Irisys .NETAPI for IRC3xxx Series (Blackfin-based) Devices

API Version 3.0.40504.01

Contents

1 Irisys .NET API for IRC3xxx series (Blackfin-based) devices **5**
- 1.1 Introduction . . . 5
- 1.2 Release notes . . . 5
  - 1.2.1 Version 3.0.40504.01 (May 2012) . . . 5
  - 1.2.2 Version 2.0.30712.01 (August 2011) . . . 6

2 Basic usage **7**
- 2.1 Linking to the API . . . 7
- 2.2 Initializing the API . . . 7
- 2.3 Connecting to a device . . . 7
  - 2.3.1 Serial connection . . . 7
  - 2.3.2 IP connection . . . 7
- 2.4 Working with device configuration . . . 8
- 2.5 Disconnecting from a device . . . 8
- 2.6 Setting IP board configuration . . . 8

3 Application notes **9**
- 3.1 Using Irisys timezones . . . 9
  - 3.1.1 Missing timezones . . . 9
- 3.2 Using device status for troubleshooting . . . 10
- 3.3 Packet timeouts . . . 10

4 Sample applications **11**
- 4.1 BlackfinAPIConsole . . . 11
- 4.2 Server samples . . . 11
  - 4.2.1 ServerSample1 . . . 11
  - 4.2.2 ServerSample2 . . . 12
  - 4.2.3 ServerSample3 . . . 12
  - 4.2.4 ServerSample4 . . . 12
- 4.3 BlackfinAPIWPFSample . . . 12

5 Count logs **13**
- 5.1 Introduction . . . 13
- 5.2 Logging period . . . 13
- 5.3 Accessing count logs . . . 13
- 5.4 Storage . . . 13
- 5.5 Resetting count logs . . . 14

6 Frequently asked questions **15**

7 Namespace documentation **16**
- 7.1 Package Irisys . . . 16
- 7.2 Package Irisys.BlackfinAPI . . . 16
- 7.3 Package Irisys.BlackfinAPI.Interfaces . . . 16
- 7.4 Package IrisysTime . . . 16

8 Class documentation **17**
- 8.1 Irisys.BlackfinAPI.Blackfin class reference . . . 17
  - 8.1.1 Detailed description . . . 19
  - 8.1.2 Constructor & destructor documentation . . . 19
    - Blackfin . . . 19
  - 8.1.3 Member function documentation . . . 19

GetCurrentCount . . . 19
GetLastNCounts . . . 19
GetCounts . . . 20
GetDeviceStatus . . . 20
GetTime . . . 20
GetUnitTime . . . 20
GetUptime . . . 20
GetUpTime . . . 20
GetUnitTimeZone . . . 21
GetMACAddress . . . 21
GetClientConfigHostname . . . 21
GetClientConfigIP . . . 21
GetDeviceName . . . 22
GetDeviceID . . . 22
GetSiteName . . . 22
GetSiteID . . . 22
GetLocaleString . . . 23
GetUserString . . . 23
GetRegisterLabels . . . 23
GetSerialNumber . . . 23
GetIPFirmwareVersion . . . 24
GetMonitorFirmwareVersion . . . 24
GetDNS1 . . . 24
GetDNS2 . . . 24
GetDNS3 . . . 25
GetClientConfigPort . . . 25
GetClientConfigPortNumber . . . 25
GetClientConfigTimeout . . . 25
GetClientConfigEnable . . . 25
GetDHCPEnabled . . . 26
GetIPAddress . . . 26
GetSubnetMask . . . 26
GetGateway . . . 26
GetCountLogPeriod . . . 27
GetNetworkChecksum . . . 27
ResetCountLogs . . . 27
ResetCurrentCount . . . 27
ResetDeviceStatus . . . 27
SetPacketTimeout . . . 28
SetCommsTimeout . . . 28
SetUnitTimeZone . . . 28
SetUnitTime . . . 28
SetTime . . . 28
SetDeviceName . . . 28
SetDeviceID . . . 29
SetSiteName . . . 29
SetSiteID . . . 29
SetLocaleString . . . 29
SetUserString . . . 30
SetDNS1 . . . 30
SetDNS2 . . . 30
SetDNS3 . . . 30
SetClientConfigIP . . . 31
SetClientConfigHostname . . . 31

SetClientConfigPort . . . 31
SetClientConfigPortNumber . . . 32
SetClientConfigTimeout . . . 32
SetClientConfigEnable . . . 32
SetCountLogPeriod . . . 32
8.1.4 Member data documentation . . . 33
MAX_PROPERTY_STRING . . . 33
MAX_DEVICEID_STRING . . . 33
8.1.5 Property documentation . . . 33
SerialNumber . . . 33
CountLogPeriod . . . 33
MACAddress . . . 33
ClientConfigIP . . . 33
ClientConfigHostname . . . 33
ClientConfigPort . . . 33
ClientConfigPortNumber . . . 33
ClientConfigTimeout . . . 33
ClientConfigEnable . . . 33
Counts . . . 33
DeviceStatus . . . 34
UnitTime . . . 34
LocaleString . . . 34
UserString . . . 34
RegisterLabels . . . 34
DeviceName . . . 34
DeviceID . . . 34
SiteName . . . 34
SiteID . . . 34
DHCPEnabled . . . 34
IPAddress . . . 34
SubnetMask . . . 34
Gateway . . . 35
IPFirmwareVersion . . . 35
MonitorFirmwareVersion . . . 35
DNS1 . . . 35
DNS2 . . . 35
DNS3 . . . 35
UpTime . . . 35
UnitTimeZone . . . 35
NetworkChecksum . . . 35
PacketTimeout . . . 35
8.2 Irisys.BlackfinAPI.BlackfinEngine class reference . . . 35
8.2.1 Detailed description . . . 36
8.2.2 Member enumeration documentation . . . 36
ConnectionErrorType . . . 36
8.2.3 Constructor & destructor documentation . . . 36
BlackfinEngine . . . 36
8.2.4 Member function documentation . . . 36
ConnectionErrorHandler . . . 36
GetAPIVersion . . . 37
StartEngine . . . 37
ShutdownEngine . . . 37
AddNewCounterEndPoint . . . 37
AddNewCounterEndPoint . . . 37

AddNewCounterEndPoint . . . 37
AddNewCounterEndPoint . . . 38
RemoveCounterEndPoint . . . 38
ClockCounters . . . 38
8.3 Irisys.BlackfinAPI.Interfaces.Count class reference . . . 38
8.3.1 Detailed description . . . 38
8.3.2 Member data documentation . . . 38
countLines . . . 38
countTime . . . 39
8.4 Irisys.BlackfinAPI.Interfaces.DeviceLogEntry class reference . . . 39
8.4.1 Detailed description . . . 39
8.4.2 Member data documentation . . . 39
errorDescription . . . 39
timestamp . . . 39
8.5 Irisys.BlackfinAPI.Interfaces.DeviceStatus class reference . . . 39
8.5.1 Detailed description . . . 39
8.5.2 Member data documentation . . . 39
m_warnList . . . 39
m_errorList . . . 39
m_infoList . . . 39
8.6 IrisysTime.IrisysTimeZone class reference . . . 40
8.6.1 Detailed description . . . 40
8.6.2 Constructor & destructor documentation . . . 40
IrisysTimeZone . . . 40
8.6.3 Property documentation . . . 40
ApplyDST . . . 40
8.6.4 Event documentation . . . 40
PropertyChanged . . . 40
8.7 IrisysTime.IrisysTimeZoneInfo class reference . . . 40
8.7.1 Detailed description . . . 41
8.7.2 Constructor & destructor documentation . . . 41
IrisysTimeZoneInfo . . . 41
8.7.3 Member function documentation . . . 41
Equals . . . 41
GetHashCode . . . 41
ToString . . . 42
8.7.4 Member data documentation . . . 42
UtcTimeZone . . . 42
8.7.5 Property documentation . . . 42
BaseUtcOffset . . . 42
SupportsDST . . . 42
TimeZoneID . . . 42
DisplayName . . . 42
TimeZones . . . 42
8.8 Irisys.BlackfinAPI.SerialNumberConverter class reference . . . 42
8.8.1 Detailed description . . . 42
8.8.2 Member function documentation . . . 42
ConvertSerialNumber . . . 42

1 Irisys .NET API for IRC3xxx series (Blackfin-based) devices

1.1 Introduction

The Irisys IRC3xxx series of People Counters provide TCP/IP connectivity via an IP master device. Irisys provides four Application Programming Interfaces (APIs) that developers can use to connect to and download count data from IRC3xxx series devices, in addition to setting and reading a number of user configurable identification strings on the device. The available APIs are;

- .NET (3.5)
- Java (1.6)
- Win32 (C++)
- Linux (C++)

Third parties can use these APIs to integrate count data from IRC3xxx series devices into their own custom applications. It is not intended to be used for the setup of counter networks or the retrieval of advanced data such as path maps or video, thus no functionality is provided for these tasks.

This documentation is intended for use by developers with a working knowledge of application programming and it is recommended to be read alongside the accompanying Sample Applications. Note that this documentation assumes devices have been installed in accordance with our recommended practices, as outlined in the accompanying Installation Guide IPU40182 and Applications Guide IPU40184. Failure to follow these practices may affect the accuracy of the count data provided by IRC3xxx series devices.

The IRC3xxx series of devices use a Blackfin processing chip. Blackfin is a Registered Trademark of Analog Devices Inc.

1.2 Release notes

1.2.1 Version 3.0.40504.01 (May 2012)

This release of the API adds support for new counter features in the latest firmware for IRC3xxx series devices, such as the ability to support up to 32 count registers.

Changelog

- Added new Irisys.BlackfinAPI.Blackfin.GetRegisterLabels() function and the associated accessor function Irisys.BlackfinAPI.Blackfin.RegisterLabels() to download the new register labels
- Functions to set property strings will now throw an ArgumentException if the length of the string to be set exceeds the maximum length of that field. Affected functions include;
  - Irisys.BlackfinAPI.Blackfin.SetDeviceName()
  - Irisys.BlackfinAPI.Blackfin.SetDeviceID()
  - Irisys.BlackfinAPI.Blackfin.SetSiteName()
  - Irisys.BlackfinAPI.Blackfin.SetSiteID()
  - Irisys.BlackfinAPI.Blackfin.SetLocaleString()
  - Irisys.BlackfinAPI.Blackfin.SetUserString()

- Added new Irisys.BlackfinAPI.Blackfin.GetNetworkChecksum() function and the associated accessor function Irisys.BlackfinAPI.Blackfin.NetworkChecksum() to create a checksum of the settings for all devices on the connected CAN network

1.2.2 Version 2.0.30712.01 (August 2011)

In this version of the API some functions have been renamed to provide consistency across all Irisys APIs for Blackfin-based devices. This is a breaking change from previous versions of the API and therefore any code based on an older version of the API may need to be updated to compile with this version. Other significant changes in this version include support for multiple (>2) count lines, Irisys time zones and improved documentation.

Changelog

- Merged BlackfinUser.dll, BlackfinCore.dll and BaseDevice.dll libraries into a single library named BlackfinUser.dll
- Moved Blackfin and BlackfinEngine classes into the Irisys.BlackfinAPI namespace, from the Blackfin_API and IrisysDevices.BlackfinCore namespaces respectively.
- Added support for Irisys time zones
- Added support for getting & setting a new 'user string' setting
- Deprecated the following functions and accessors, each of which has been replaced with an equivilent function which is consistent with the other Blackfin APIs;
  - Irisys.BlackfinAPI.Blackfin.GetTime(), which has been replaced with Irisys.BlackfinAPI.Blackfin.GetUnitTime()
  - Irisys.BlackfinAPI.Blackfin.SetTime(), which has been replaced with Irisys.BlackfinAPI.Blackfin.SetUnitTime()
  - Irisys.BlackfinAPI.Blackfin.GetClientConfigPortNumber(), which has been replaced with Irisys.BlackfinAPI.Blackfin.GetClientConfigPort()
  - Irisys.BlackfinAPI.Blackfin.SetClientConfigPortNumber(), which has been replaced with Irisys.BlackfinAPI.Blackfin.SetClientConfigPort()
  - Irisys.BlackfinAPI.Blackfin.GetUptime(), which has been replaced with Irisys.BlackfinAPI.Blackfin.GetUpTime()
  - Irisys.BlackfinAPI.Blackfin.ClientConfigPortNumber, which has been replaced with Irisys.BlackfinAPI.Blackfin.ClientConfigPort
- Made a lot of classes internal rather than public. This should not affect user code
- Consolidated and improved SampleServer sample application to aid understanding
- Added Irisys.BlackfinAPI.SerialNumberConverter class to convert between numeric and alpha-numeric serial number representations

2 Basic usage

This section provides an introduction to the basic usage of the %Irisys API (.NET) to connect to a device and perform some simple tasks with it. For more detailed information about the functions named in this section and other functions available in the API refer to the included class documentation.

2.1 Linking to the API

You need to include BlackfinUser.dll as a reference in your .NET project. This will make all of the user classes available from within the Irisys.BlackfinAPI and IrisysTime namespaces. Visual Studio automatically copies the DLL into the executable path.

2.2 Initializing the API

Before attempting to use any of the functions or classes provided with this API you must first initialise it by creating an object of type Irisys.BlackfinAPI.BlackfinEngine and calling the Irisys.BlackfinAPI.BlackfinEngine.StartEngine() member function. This object will be used by your application to create and destroy connections to devices.

Upon shutdown your application should clean up your engine by calling the Irisys.BlackfinAPI.BlackfinEngine.ShutdownEngine() member function. Note that any connected Irisys.BlackfinAPI.Blackfin objects should first be disconnected by passing them to the Irisys.BlackfinAPI.BlackfinEngine.RemoveCounterEndPoint() member function. After shutting down the engine your application must not use any other functions or classes provided with this API.

2.3 Connecting to a device

The API supports connecting to devices by either serial or TCP/IP, where supported by the device. For either type of connection you must first create an object of type Irisys.BlackfinAPI.Blackfin using the default constructor.

2.3.1 Serial connection

To connect via a serial COM port you should create an object of type System.IO.Ports.SerialPort on the desired port. You should then call the Irisys.BlackfinAPI.BlackfinEngine.AddNewCounterEndPoint() function on your initialized Irisys.BlackfinAPI.BlackfinEngine object, passing as the parameters the Irisys.BlackfinAPI.Blackfin object you created earlier and your new serial port object.

2.3.2 IP connection

There are two mechanisms for creating connections to devices using TCP/IP, which are only possible on devices which feature an IP board. Your application can connect to the device directly by creating an object of type System.Net.Sockets.Socket and connecting it to the devices IP address.

Alternatively your application can recieve an incoming connection from a device with client connection mode enabled, which can be configured via the People Counter Setup Tool or by using the SetClientConfig* family of commands in the API. An object of type System.Net.Sockets.TcpListener can be used to listen for these connections, as demonstrated in the sample server applications provided.

Regardless of how you created your IP connection you should call the Irisys.BlackfinAPI.BlackfinEngine.AddNewCounterEndPoint() function of your initialized Irisys.BlackfinAPI.BlackfinEngine object, passing in the Irisys.BlackfinAPI.Blackfin object created earlier and your connected socket.

2.4 Working with device configuration

Once an Irisys.BlackfinAPI.Blackfin object is connected to a device there are many API functions that can be used to interact with the device. Many of these functions relate to getting and setting device configuration items such as the device id and device name. The usage of these functions are all broadly similar to the example below demonstrating how to get and set the site name configuration item on a connected device.

```
*     // In this code bfin is a connected instance of the \bfclass class
*
*     // Getting the site name from the device
*     if ( bfin.GetSiteName() )
*       Console.WriteLine( "Device site name is " + bfin.SiteName );
*     else
*       Console.WriteLine( "Failed to get site name" );
*
*     // Setting the site name on the device
*     if ( bfin.SetSiteName("My Site 1") )
*       Console.WriteLine( "Successfully set new site name on the device" );
*     else
*       Console.WriteLine( "Failed to set site name" );
*
```

The Irisys.BlackfinAPI.Blackfin.GetSiteName() function initiates a blocking communications call to the connected device to query its configured site name string, returning true upon success or false after a timeout or any other failure. If successful the site name string configured on the device is available via the Irisys.BlackfinAPI.Blackfin.SiteName property.

Similarly the Irisys.BlackfinAPI.Blackfin.SetSiteName() initiates a blocking communications call to the connected device to write the new site name string to its configuration, returning true upon success or false after a timeout or any other failure.

2.5 Disconnecting from a device

To disconnect a Irisys.BlackfinAPI.Blackfin object from a serial port or IP socket you should pass it to the Irisys.BlackfinAPI.BlackfinEngine.RemoveCounterEndPoint() function of your initialized Irisys.BlackfinAPI.BlackfinEngine object. Your application must disconnect all devices in this way before shutting down the engine as part of its general cleanup.

2.6 Setting IP board configuration

Whilst it is possible to call the SetClientConfig* and SetDNS* families of functions whilst connected to a device by TCP/IP it is **NOT** recommended as this will cause the IP board to be reset and drop the connection. If your application does this your Irisys.BlackfinAPI.Blackfin will be unable to communicate with the device and you should disconnect it and destroy the object before, if necessary, creating a new connection as outlined above. If a connection to a device connected in client connection mode is dropped in this way you may be unable to connect to that device again until the TCP/IP timeout occurs, which may take up to 10 minutes. It is recommended that you connect to a device via serial if you wish to call any of these functions. Note that the GetClientConfig* and GetDNS* families of functions are not affected by this.

3 Application notes

3.1 Using Irisys timezones

As of API version 2.0.30712.01, it is now possible to get and set the time zone in which a device resides - the same setting which is available in the People Counter Setup Tool and Validation Tool. This setting does not affect the Irisys.BlackfinAPI.Blackfin.GetUnitTime() function which always returns a value in UTC time.

The time zone setting is restricted to a (large) range of time zones which correspond to Windows time zones ID's. The function IrisysTime.IrisysTimeZoneInfo.TimeZones enumerates the set of time zones available. Each IrisysTime.IrisysTimeZone object has a base UTC offset and a Boolean value representing whether or not the time zone supports Daylight Savings Time (DST). When setting the time zone, the appropriate IrisysTime.IrisysTimeZoneInfo object must be passed along with a boolean representing whether or not DST is to be applied or not. Note the distinction between the user information returned by IrisysTime.IrisysTimeZone.ApplyDST and the API information returned by IrisysTime.IrisysTimeZoneInfo.SupportsDST.

If DST is to be taken into account when converting a UTC time to a local time then it is necessary to have more information than the API provides - the base UTC offset must be adjusted by a set of rules dependant on the particular time zone. These rules are non-trivial for many zones and are typically represented in a database. This information is available in the Windows registry, or by using the System.TimeZoneInfo class.

General procedure for calculating local time:

- Retrieve UTC time from device using Irisys.BlackfinAPI.Blackfin.GetUnitTime()
- Retrieve time zone from device using Irisys.BlackfinAPI.Blackfin.GetUnitTimeZone()
- If not applying daylight savings rules, add the time zone base UTC offset to the UTC time.
- If applying daylight savings rules:
  - Query database (e.g. the Windows Registry) for daylight savings rules
  - Apply rules to the UTC time to generate appropriate UTC offset for that point in time
  - Add UTC offset to the UTC time to get adjusted local time

3.1.1 Missing timezones

Some of the time zone IDs contained within the Irisys API do not exist on some versions of Microsoft Windows when the latest time zone updates have not been installed, therefore you are advised not to assume the registry keys containing the time zone information exist on a given target machine.

You can add the missing time zones on Microsoft Windows NT 5.0 (XP, Server 2000) and newer by installing the latest time zone update via Windows Update or by downloading the latest time zone update package from the Microsoft Support website.

3.2 Using device status for troubleshooting

The Irisys.BlackfinAPI.Blackfin.GetDeviceStatus() function allows you to query the diagnostic log of the connected device, the results of which will be available via the Irisys.BlackfinAPI.Blackfin.-DeviceStatus property. This can be used to determine if the device is in an error state or is issuing any warning messages, as well as informative messages such as the last reset time.

Note that IRC3xxx series devices will reset a device every minute in the event that a fatal error condition (such as array failure) has occurred. This ensures the device never becomes unresponsive, however it will cause any communications calls to be more likely to fail, especially those which take a long time to complete such as downloading the count log.

3.3 Packet timeouts

Communications calls made by an Irisys.BlackfinAPI.Blackfin instance are marked as having failed if no response is received within the configured timeout period, which defaults to 5 seconds. You can query the current timeout period using the Irisys.BlackfinAPI.Blackfin.PacketTimeout property. For the majority of API functions this timeout is the maximum time the function will block for before returning a result, with the exceptions to this rule being;

- Irisys.BlackfinAPI.Blackfin.GetCounts()
- Irisys.BlackfinAPI.Blackfin.GetLastNCounts()
- Irisys.BlackfinAPI.Blackfin.GetDeviceStatus()

If you are connecting to a device on a high latency network it may be useful to increase the timeout period using the Irisys.BlackfinAPI.Blackfin.SetPacketTimeout() function to allow the API more time to wait for responses from that device, however this has the side effect of increasing the delay before a blocking API function will return a failure when the connection has been lost. One potential strategy is to count the number of failed API calls and, upon reaching a set threshold, raise the timeout value to try and work around network latency issues.

4 Sample applications

This section provides a brief description of the sample applications included with the .NET version of the API. If you have received this document without the sample applications listed below please contact your supplier.

4.1 BlackfinAPIConsole

A simple console application which demonstrates the correct procedure for connecting directly to a device using a TCP/IP socket and allows the user to test the various functions available through the API.

This application is a useful way of exploring the functionality available through the API and performing simple tasks on a device and could easily be modified to work with batch or powershell scripts if desired, allowing the API to be used indirectly with almost any programming language.

The entry point for the application creates a BlackfinEngine object and initializes it so it is ready to create connections. It then prompts the user for an IP address of a Blackfin device and creates a TCP/IP socket to connect to that address on port 4505. A Blackfin object is created and connected to the socket by passing both to the AddNewCounterEndPoint function of the BlackfinEngine. If the connection is successful it drops into a command processing loop until the device is disconnected, at which point the engines RemoveCounterEndPoint is called to clean up the connection and the application will prompt the user for a new IP address to connect to.

Whilst in the command processing loop any of the commands available in the API can be tested by typing in their name when prompted and providing appropriate parameters on request. The help command can be used to output a list of all available commands and a short description of that command. Note that using the SetClientConfig* or SetDNS* commands will disconnect the application from a device and you must reconnect manually if desired.

4.2 Server samples

A selection of sample server applications which listen for and accept client connections from Blackfin devices, upon which they execute a series of API calls. Each of the samples represents a different usage scenario.

4.2.1 ServerSample1

This server sample adds all incoming connections to a queue for processing, ensuring no incoming connections are missed whilst the previous one is being processed. The data thread checks to see if there are any Blackfin objects in its queue and, if so, removes the first item. It then executes a series of API calls on that object before disconnecting it from the server. This usage scenario ensures only one device is actively communicating with the application at any given time, but allows for other devices to connect and wait to be processed.

4.2.2 ServerSample2

This server sample spawns a thread for each connection it recieves, which then creates a Blackfin object for the connection and associates them via the BlackfinEngine. This allows the main thread to continue accepting new connections whilst existing ones are being processed. Once all the API functions have been executed the Blackfin object is removed from the engine, the connection closed and the thread stops. This usage scenario allows multiple devices to communicate with the application at the same time, each in their own thread.

4.2.3 ServerSample3

This server sample spawns a thread for each connection it recieves, which then creates a BlackfinWrapper object for the connection. This wrapper around the Blackfin class provides functions for connecting the underlying Blackfin object to a socket via the BlackfinEngine and executing a variety of API calls. All of the functions in the wrapper handle comms errors internally, allowing the main application code to be written more cleanly. Once all the API functions have been executed the BlackfinWrapper is deleted, resulting in the underlying Blackfin object being removed from the engine and the connection closed, at which point the thread stops. This usage scenario allows multiple devices to communicate with the application at the same time, each in their own thread.

4.2.4 ServerSample4

This server sample is almost identical to ServerSample3, except that it does not close the connection once all of the API functions have been executed. Instead the thread calls the same set of API functions every 30 seconds, maintaining a persistent connection to the Blackfin device. This usage scenario scenario allows multiple devices to communicate with the application at the same time, each in their own thread and maintaining persistent connections.

4.3 BlackfinAPIWPFSample

This is a simple GUI application built with the WPF toolkit which allows a user to connect to an IRC3xxx series device via IP or serial and execute a variety of API commands upon it.

Like the BlackfinAPIConsole sample application this provides another way of exploring the functionality available via the API with the added ease of a GUI in place of the command line interface. This application can easily be used to configure some basic settings on a device if desired.

To connect to a device either type its IP address into the text box below the Connect button and click Connect or select a serial COM# port from the list and click the Serial Connect button. If successful the buttons background will turn green and the rest of the GUI will be enabled, otherwise the buttons background will turn red and an error message will be shown. To connect to another device click on the Disconnect or Serial Disconnect button (depending on how you connected to the device) and either enter a new IP address or select a new serial COM# number to connnect to and click the appropriate button.

Once connected the other buttons in the application can be used to get and set all of the configuration items which are available via the API and perform other functions available via the API, such as downloading count log entries from the device. The success or failure or each function call is indicated by the background of the button turning green or red and, if getting information, the GUI will be updated with the information retrieved.

5 Count logs

5.1 Introduction

This section describes how Irisys IRC3xxx series devices handle count logs internally and how you can interact with them using the API. A count log is a record of the count values for each configured register at the time the log was written to the device's non-volatile memory.

5.2 Logging period

The frequency with which count logs are created is determined by the CountLogPeriod setting which specifies, in seconds, the interval between each count log. The interval is based on the number of seconds since the start of the day to provide predictable logging times, hence the default logging interval of 900 seconds (15 minutes) will create count logs at 0000, 0015, 0030, etc.

You can use the Irisys.BlackfinAPI.Blackfin.GetCountLogPeriod() API function and the associated accessor function Irisys.BlackfinAPI.Blackfin.CountLogPeriod() to read the current value of this setting from the device. The Irisys.BlackfinAPI.Blackfin::SetCountLogPeriod() API function or the People Counter Setup Tool can be used to change the value of this setting. Any changes to this setting will take effect immediately without any further action required.

5.3 Accessing count logs

There are two functions in the API for retrieving count logs from a device, Irisys.BlackfinAPI.Blackfin.GetCounts() and Irisys.BlackfinAPI.Blackfin.GetLastNCounts(). The first of these allows you to retrieve all count logs which fall into a specified time period, whilst the second retrieves a specified number of the most recent log entries.

Regardless of which function you call, the retrieved count log entries are available via the Irisys.BlackfinAPI.Blackfin.Counts property, which returns a List containing an Irisys.BlackfinAPI.Interfaces.Count instance for each count log entry downloaded from the device. This class in turn contains a UTC timestamp representing the time the log entry was recorded and a List of unsigned integers (32bits) which contain the count value for each configured counter. The size of this list depends on the number of count registers configured at the time the log entry was recorded and may not be consistent between all of the log entries returned if the device configuration was changed during the log period retrieved.

Note that subsequent calls to any of the Irisys.BlackfinAPI.Blackfin.GetCounts(), Irisys.BlackfinAPI.Blackfin.GetLastNCounts() or Irisys.BlackfinAPI.Blackfin.GetCurrentCount() functions will overwrite the downloaded count data with the result of that function call.

5.4 Storage

Count logs are written to the device's non-volatile flash memory and therefore are not lost even if the power to the device is removed. The number of count log entries which can be stored on the device depends on the number of count registers which have been configured.

Due to the nature of the flash memory used on the device the only way to erase data is to erase the entire sector. Therefore the count logs are split between two sectors, with the oldest sector being erased when the newer sector is full. Assuming you do not reset the count logs at any point this means that the minimum number of stored count logs (after both flash sectors have been filled at least once) is half of the maximum theoretical

capacity of the device and the number of counts available via the API will be any amount between the minimum and maximum.

The maximum and minimum storage capacities for a range of configured registers are given in the table below, along with the minimum number of days worth of logging this represents at 5, 15 and 60 minute logging intervals.

Registers	Minimum Capacity	Maximum Capacity	5 Minute Interval (Min)	15 Minute Interval (Min)	60 Minute Interval (Min)
2	4680	9361	16 Days	**48 Days**	195 Days
4	2978	5957	10 Days	31 Days	124 Days
8	1724	3448	5 Days	17 Days	71 Days
16	936	1872	3 Days	9 Days	39 Days
32	489	978	1 Day	5 Days	20 Days

The default logging period of 15 minutes with 2 enabled count registers will have a minimum capacity of 48 days worth of count log entries.

5.5 Resetting count logs

You can use the API to reset the count log on an IRC3xxx series device by calling the Irisys.-BlackfinAPI.Blackfin.ResetCountLogs() function, which will permanently erase all count log entries from the device. Note that flash memory has a finite lifespan and excessive use of this function could reduce the lifespan of your devices. Rather than resetting the count logs your application should keep track of the most recent count log entry retrieved from the device and use this as the starting point when requesting new data.

6 Frequently asked questions

- **What is the baud rate (speed) of an IP connection?**

  An Irisys IRC3xxx device can have a link speed of 10Mbps or 100Mbps, however the maximum data baud rate is limited to 1Mbps

- **What is the baud rate (speed) of a serial connection?**

  The baud rate of a serial connection to an Irisys IRC3xxx device is 115200bps

- **Which TCP/IP ports are used to communicate with an Irisys IRC3xxx device?**

  Port 4505 is used to establish direct IP connections to the device.

  Port 80 is used to connect to the web interface on the device and configure it using the built in People Counter Setup Tool (PCST) software.

  By default port 5000 is used by devices for outgoing connections when client connection mode is enabled, however this can be configured via the web interface.

- **What is the default IP address of a device?**

  The factory default IP address is 192.168.0.10, using a subnet mask of 255.255.255.0

- **What is the maximum value of a count line?**

  The count registers on the device are 32 bits, giving a maximum value of over 4.2 billion, which is reset to 0 each time the device is powered off and on again.

- **What is the lifespan of the flash memory storage on the device?**

  The flash memory used on the device has a rated minimum lifespan of 100,000 erase / write cycles, which equates to roughly 26,000 years of logging at a 15 minute interval. Resetting the count log files will cause additional erase / write cycles to occur and will reduce the useful life of the device, therefore you should avoid doing this unless absolutely necessary. For more information about the count log files on the device see the Count Logs section.

7 Namespace documentation

7.1 Package Irisys

Namespaces

- package BlackfinAPI

7.2 Package Irisys.BlackfinAPI

Namespaces

- package Interfaces

Classes

- class BlackfinEngine
- class Blackfin
- class SerialNumberConverter

7.3 Package Irisys.BlackfinAPI.Interfaces

Classes

- class Count
- class DeviceLogEntry
- class DeviceStatus

7.4 Package IrisysTime

Classes

- class IrisysTimeZone
- class IrisysTimeZoneInfo

8 Class documentation

8.1 Irisys.BlackfinAPI.Blackfin class reference

Public Member Functions

- Blackfin ()
- bool ResetCountLogs ()
- bool ResetCurrentCount ()
- bool ResetDeviceStatus ()

Setters

- bool SetPacketTimeout (int milliseconds)
- bool SetCommsTimeout (int milliseconds)
- bool SetUnitTimeZone (IrisysTimeZone timezone)
- bool SetUnitTime (DateTime tm)
- bool SetTime (DateTime t)
- bool SetDeviceName (string deviceName)
- bool SetDeviceID (string deviceID)
- bool SetSiteName (string sitename)
- bool SetSiteID (string siteid)
- bool SetLocaleString (string locale)
- bool SetUserString (string user)
- bool SetDNS1 (IPAddress dns1)
- bool SetDNS2 (IPAddress dns2)
- bool SetDNS3 (IPAddress dns3)
- bool SetClientConfigIP (IPAddress ipAddress)
- bool SetClientConfigHostname (string hostname)
- bool SetClientConfigPort (ushort portNumber)
- bool SetClientConfigPortNumber (ushort portNumber)
- bool SetClientConfigTimeout (uint timeout)
- bool SetClientConfigEnable (bool enable)
- bool SetCountLogPeriod (int countLogPeriod)

Public Attributes

- const int MAX_PROPERTY_STRING = 64
- const int MAX_DEVICEID_STRING = 160

Properties

- int SerialNumber `[get, set]`
- int CountLogPeriod `[get, set]`
- string MACAddress `[get, set]`
- IPAddress ClientConfigIP `[get, set]`
- string ClientConfigHostname `[get, set]`
- ushort ClientConfigPort `[get, set]`
- ushort ClientConfigPortNumber `[get]`
- uint ClientConfigTimeout `[get, set]`
- bool ClientConfigEnable `[get, set]`
- List< Count > Counts `[get, set]`
- DeviceStatus DeviceStatus `[get, set]`
- DateTime UnitTime `[get, set]`
- string LocaleString `[get, set]`

- string UserString [get, set]
- List< string > RegisterLabels [get, set]
- string DeviceName [get, set]
- string DeviceID [get, set]
- string SiteName [get, set]
- string SiteID [get, set]
- bool DHCPEnabled [get, set]
- IPAddress IPAddress [get, set]
- IPAddress SubnetMask [get, set]
- IPAddress Gateway [get, set]
- string IPFirmwareVersion [get, set]
- string MonitorFirmwareVersion [get, set]
- IPAddress DNS1 [get, set]
- IPAddress DNS2 [get, set]
- IPAddress DNS3 [get, set]
- UInt64 UpTime [get, set]
- IrisysTimeZone UnitTimeZone [get, set]
- String NetworkChecksum [get, set]
- int PacketTimeout [get]

Getters

- bool GetCurrentCount ()
- bool GetLastNCounts (uint n)
- bool GetCounts (DateTime start, DateTime end)
- bool GetDeviceStatus ()
- bool GetTime ()
- bool GetUnitTime ()
- bool GetUptime ()
- bool GetUpTime ()
- bool GetUnitTimeZone ()
- bool GetMACAddress ()
- bool GetClientConfigHostname ()
- bool GetClientConfigIP ()
- bool GetDeviceName ()
- bool GetDeviceID ()
- bool GetSiteName ()
- bool GetSiteID ()
- bool GetLocaleString ()
- bool GetUserString ()
- bool GetRegisterLabels ()
- bool GetSerialNumber ()
- bool GetIPFirmwareVersion ()
- bool GetMonitorFirmwareVersion ()
- bool GetDNS1 ()
- bool GetDNS2 ()
- bool GetDNS3 ()
- bool GetClientConfigPort ()
- bool GetClientConfigPortNumber ()
- bool GetClientConfigTimeout ()
- bool GetClientConfigEnable ()

- bool GetDHCPEnabled ()
- bool GetIPAddress ()
- bool GetSubnetMask ()
- bool GetGateway ()
- bool GetCountLogPeriod ()
- bool GetNetworkChecksum ()

8.1.1 Detailed description

Class used for accessing data and settings from an Irisys IP device

A Blackfin instance must be added to a BlackfinEngine along with a socket or serial port in order to retrieve or set data on a remote device.

Note that when connected to a non-master device via serial many of the comms function calls will fail. This is because some of them require the require the IP board (master only) to be present. This includes all functions with the ClientConfig prefix.

8.1.2 Constructor & destructor documentation

Irisys.BlackfinAPI.Blackfin.Blackfin ( )

Creates a new blank Blackfin object, not associated with any connection.

8.1.3 Member function documentation

bool Irisys.BlackfinAPI.Blackfin.GetCurrentCount ( )

Get the live count line values from the device. These values are not written to the count log and therefore may not be the same as the most recent entry returned by the GetCounts() function. Upon successful completion the count line values will be available via the Counts variable.

See also

Counts

Returns

True on success, false on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetLastNCounts (uint *n*)

Get the latest N count log entries from the device. Upon successful completion the count log entries will be available via the Counts variable.

Note

The number of count log entries retrieved may be less than N if there are less than N entries in the device count log

Parameters

n	Number of count log entries to retrieve

See also

Counts

Returns

True on success, false on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetCounts (DateTime *start,* DateTime *end*)

Gets the count log entries from the device which fall within the specified time range. Upon successful completion the count log entries will be available via the Counts variable.

Parameters

start	UTC time of the earliest count log entry to retrieve
end	UTC time of the latest count log entry to retrieve

See also

Counts

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetDeviceStatus ( )

Get the device status logs since the last boot up or device log reset. Upon successful completion the device status logs will be available via the DeviceStatus variable.

Note

This call may take longer than the configured packet timeout

See also

DeviceStatus

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetTime ( )

bool Irisys.BlackfinAPI.Blackfin.GetUnitTime ( )

Get the current UTC time of the devices internal clock, which is not affected by the configured timezone. Upon successful completion the devices UTC time will be available via the UnitTime variable.

See also

UnitTime

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetUptime ( )

bool Irisys.BlackfinAPI.Blackfin.GetUpTime ( )

Get the elapsed time, in seconds, since the device was booted. Upon successful completion the uptime will be available via the UpTime variable.

See also

UpTime

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetUnitTimeZone ( )

Get the timezone configured on the device. Note that this has no functional impact on the device itself, which will always use UTC time. Upon successful completion the timezone configured on the device will be available via the UnitTimeZone variable.

Note

The timezone can be configured using the People Counter Setup Tool

UnitTimeZone

Returns

True on success, false if the devices firmware is too old or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetMACAddress ( )

Get the MAC address of the device, which must feature an IP board. Upon successful completion the devices MAC address will be available via the MACAddress variable.

See also

MACAddress

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetClientConfigHostname ( )

Get the client connection hostname configured on the device, which must feature an IP board. This setting specifies the hostname the device will attempt to connect to when client connection mode is enabled. Upon successful completion the client connection hostname configured on the device will be available via the ClientConfigHostname variable.

Note

If both an IP address and hostname are specified for client connection mode the IP address is used

See also

ClientConfigHostname

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetClientConfigIP ( )

Get the client connection IP address configured on the device, which must feature an IP board. This setting specifies the IP address the device will attempt to connect to when client connection mode is enabled. Upon successful completion the client connection IP address configured on the device will be available via the ClientConfigIP variable.

Note

If both an IP address and hostname are specified for client connection mode the IP address is used

See also

ClientConfigIP

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetDeviceName ( )

Get the device name string from the device. Upon successful completion the device name will be available via the DeviceName variable.

See also

DeviceName

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetDeviceID ( )

Get the device ID string from the device. Upon successful completion the device ID will be available via the DeviceID variable.

See also

DeviceID

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetSiteName ( )

Get the site name string from the device. Upon successful completion the site name will be available via the SiteName variable.

See also

SiteName

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetSiteID ( )

Get the site ID string from the device. Upon successful completion the site ID name will be available via the SiteID variable.

See also

SiteID

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetLocaleString ( )

Get the locale string from the device. Upon successful completion the locale string will be available via the LocaleString variable.

Note

This value is not linked to the unit timezone and has no functional impact on the device

See also

LocaleString

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetUserString ( )

Get the user string from the device. Upon successful completion the user string will be available via the UserString variable.

See also

UserString

Returns

True on success, false if the devices firmware is too old or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetRegisterLabels ( )

Retrieve the labels for the enabled count registers from the device. Upon successful completion the retrieved labels will be available via the RegisterLabels variable This command was introduced in the 3.0.x series of APIs

Minimum DSP Firmware Version: 454

See also

RegisterLabels

Returns

True on success, false if the devices firmware is too old or a comms failure occurs

bool Irisys.BlackfinAPI.Blackfin.GetSerialNumber ( )

Get the unique serial number from the device. Upon successful completion the serial number will be available via the SerialNumber variable.

See also

SerialNumber

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetIPFirmwareVersion ( )

Get the version of IP firmware installed on the device, which must feature an IP board. Upon successful completion the IP firmware version of the device will be available via the IPFirmwareVersion variable.

See also

IPFirmwareVersion

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetMonitorFirmwareVersion ( )

Get the version of firmware installed on the device. Upon successful completion the firmware version of the device will be available via the MonitorFirmwareVersion variable.

See also

MonitorFirmwareVersion

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetDNS1 ( )

Get the primary DNS server for the device, which must feature an IP board. Upon successful completion the primary DNS server for the device will be available via the DNS1 variable.

See also

DNS1

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetDNS2 ( )

Get the secondary DNS server for the device, which must feature an IP board. Upon successful completion the secondary DNS server for the device will be available via the DNS2 variable.

See also

DNS2

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetDNS3 ( )

Get the tertiary DNS server for the device, which must feature an IP board. Upon successful completion the tertiary DNS server for the device will be available via the DNS3 variable.

See also

DNS3

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetClientConfigPort ( )

Get the client connection port configured on the device, which must feature an IP board. This setting specifies the port the device will attempt to connect to when client connection mode is enabled. Upon successful completion the client connection port configured on the device will be available via the ClientConfigPort variable.

See also

ClientConfigPort

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetClientConfigPortNumber ( )

bool Irisys.BlackfinAPI.Blackfin.GetClientConfigTimeout ( )

Get the client connection timeout configured on the device, which must feature an IP board. This setting specifies the timeout, in seconds, between client connection attempts by the device when client connection mode is enabled. Upon successful completion the client connection timeout configured on the device will be available via the ClientConfigTimeout variable.

See also

ClientConfigTimeout

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetClientConfigEnable ( )

Get the client connection state configured on the device, which must feature an IP board. This setting specifies whether the device will attempt to establish an outgoing connection to the IP address or hostname configured on the device. Upon successful completion the client connection state configured on the device will be available via the ClientConfigEnable variable.

See also

ClientConfigEnable

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetDHCPEnabled ( )

Get the DHCP state of the device, which must feature an IP board. This setting determines if the device obtains an IP address from a DHCP server or uses a statically configured address. Upon successful completion the DHCP state of the device will be available via the DHCPEnabled variable.

See also

DHCPEnabled

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetIPAddress ( )

Get the statically configured IP address of the device, which must feature an IP board. This setting determines the IP address of a device when it is not using DHCP. Upon successful completion the IP address of the device will be available via the IPAddress variable.

See also

IPAddress

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetSubnetMask ( )

Get the statically configured subnet mask of the device, which must feature an IP board. This setting determines the subnet mask of a device when it is not using DHCP. Upon successful completion the subnet mask of the device will be available via the SubnetMask variable.

See also

SubnetMask

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetGateway ( )

Get the statically configured gateway of the device, which must feature an IP board. This setting determines the gateway of a device when it is not using DHCP. Upon successful completion the gateway of the device will be available via the Gateway variable.

See also

Gateway

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.GetCountLogPeriod ( )

Get the count log period currently configured on the device, which specifies the interval, in seconds, between count log entries being written to the devices memory. Upon successful completion the configured count log period will be available via the CountLogPeriod variable.

See also

CountLogPeriod

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.GetNetworkChecksum ( )

Generates a checksum of the current configuration settings of all devices on the CAN network of the connected device, which can be used to detect whether any configuration changes have taken place on the network between two subsequent calls to this function. Upon successful completion the network checksum will be available via the NetworkChecksum() accessor function.

Note

This checksum may also be affected by firmware upgrades to any device on the CAN network

This command was introduced in the 3.0.x series of APIs

See also

NetworkChecksum()

Returns

True on success, false if a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.ResetCountLogs ( )

Empty the count log stored on the device. This action cannot be undone and will not affect the live count line values.

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.ResetCurrentCount ( )

Reset the live count line values on the device. This action cannot be undone and does not affect existing count log entries stored on the device.

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.ResetDeviceStatus ( )

Clear all entries in the device status logs. This action cannot be undone.

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.SetPacketTimeout (int *milliseconds*)

Set the packet timeout value for the API to use when communicating with the connected device, which specifies how long it will wait for a response before returning a failure. Longer values can help reduce failures on high latency connections but will also cause Get* and Set* function calls to block for longer whilst they wait for a response when the connection has been lost.

See also

PacketTimeout

Parameters

milliseconds	Timeout value, in milliseconds, to use during communications with the device

bool Irisys.BlackfinAPI.Blackfin.SetCommsTimeout (int *milliseconds*)

bool Irisys.BlackfinAPI.Blackfin.SetUnitTimeZone (IrisysTimeZone *timezone*)

Set the time zone on the device. Note that this has no functional impact on the device itself, which will always use UTC time internally.

Note

The time zone can also be configured using the People Counter Setup Tool

Parameters

timezone	Irisys time zone to set on the device

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.SetUnitTime (DateTime *tm*)

Sets the internal clock on the device to the specified time, which should be in UTC
Parameters

tm	The UTC time to set on the device

Returns

True on success, False on comms error or timeout

bool Irisys.BlackfinAPI.Blackfin.SetTime (DateTime *t*)

bool Irisys.BlackfinAPI.Blackfin.SetDeviceName (string *deviceName*)

Set the device name string on the device to the specified value
Parameters

deviceName	Device name string to set on the device, up to 63 characters long

Returns

True on success, False on comms error or timeout

Exceptions

ArgumentException	Thrown if the specified string exceeds the maximum length

bool Irisys.BlackfinAPI.Blackfin.SetDeviceID (string *deviceID*)

Set the device ID string on the device to the specified value
Parameters

deviceID	Device ID string to set on the device, up to 159 characters long

Returns

True on success, False on comms error or timeout

Exceptions

ArgumentException	Thrown if the specified string exceeds the maximum length

bool Irisys.BlackfinAPI.Blackfin.SetSiteName (string *sitename*)

Set the site name string on the device to the specified value
Parameters

sitename	Site name string to set on the device, up to 63 characters long

Returns

True on success, False on comms error or timeout

Exceptions

ArgumentException	Thrown if the specified string exceeds the maximum length

bool Irisys.BlackfinAPI.Blackfin.SetSiteID (string *siteid*)

Set the site ID string on the device to the specified value
Parameters

siteid	Site ID string to set on the device, up to 63 characters long

Returns

True on success, False on comms error or timeout

Exceptions

ArgumentException	Thrown if the specified string exceeds the maximum length

bool Irisys.BlackfinAPI.Blackfin.SetLocaleString (string *locale*)

Set the locale string on the device to the specified value

Parameters

locale	Locale string to set on the device, up to 63 characters long

Returns

True on success, False on comms error or timeout

Exceptions

ArgumentException	Thrown if the specified string exceeds the maximum length

bool Irisys.BlackfinAPI.Blackfin.SetUserString (string *user*)

Set the user string on the device to the specified value
Parameters

user	User string to set on the device, up to 63 characters long

Returns

True on success, False on comms error or timeout

Exceptions

ArgumentException	Thrown if the specified string exceeds the maximum length

bool Irisys.BlackfinAPI.Blackfin.SetDNS1 (IPAddress *dns1*)

Set the primary DNS server for the device to use when resolving hostnames, which must feature an IP board. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.
Parameters

dns1	Primary DNS server IP address to set on this device

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetDNS2 (IPAddress *dns2*)

Set the secondary DNS server for the device to use when resolving hostnames, which must feature an IP board. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.
Parameters

dns2	Secondary DNS server IP address to set on this device

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetDNS3 (IPAddress *dns3*)

Set the tertiary DNS server for the device to use when resolving hostnames, which must feature an IP board. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.

Parameters

dns3	Tertiary DNS server IP address to set on this device

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetClientConfigIP (IPAddress *ipAddress*)

Set the client connection IP address on the device, which must feature an IP board. This setting specifies the IP address the device will attempt to connect to when client connection mode is enabled. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.

Note

Set this value to 0.0.0.0 if you want to specify a hostname to connect to instead

Parameters

ipAddress	The client connection IP address to set on the device

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetClientConfigHostname (string *hostname*)

Set the client connection hostname on the device, which must feature an IP board. This setting specifies the hostname the device will attempt to connect to when client connection mode is enabled. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.

Note

If a client connection IP address other than 0.0.0.0 is specified this setting has no effect

Parameters

hostname	The client connection hostname to set on the device

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetClientConfigPort (ushort *portNumber*)

Set the client connection port number on the device, which must feature an IP board. This setting specifies the port the device will attempt to connect to when client connection mode is enabled. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.

Parameters

portNumber	The client connection port number to set on the device

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetClientConfigPortNumber (ushort *portNumber*)

bool Irisys.BlackfinAPI.Blackfin.SetClientConfigTimeout (uint *timeout*)

Set the client connection timeout on the device, which must feature an IP board. This setting specifies the timeout, in seconds, between client connection attempts by the device when client connection mode is enabled. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.

Setting IP board configuration

Parameters

timeout	The client connection timeout to set on the device, in seconds

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetClientConfigEnable (bool *enable*)

Enable or disable client connection mode on the device, which must feature an IP board. If this setting is enabled the device will regularly attempt to establish an outgoing client connection to the IP address or hostname specified in the client connection settings. Please refer to the notes on Setting IP board configuration before using this function with a TCP/IP connection.

Parameters

enable	Client connection enabled state to set on the device

See also

SetClientConfigIP, SetClientConfigHostname, SetClientConfigPort, SetClientConfigTimeout

Returns

True on success, false if the device does not have an IP board or a timeout or other comms error occurred

bool Irisys.BlackfinAPI.Blackfin.SetCountLogPeriod (int *countLogPeriod*)

Set the count log period on the device which specifies the interval, in seconds, between count log entries being written to the devices memory.

Parameters

countLog-Period	The count log period to set on the device, in seconds

Returns

True on success, False on comms error or timeout

8.1.4 Member data documentation

const int Irisys.BlackfinAPI.Blackfin.MAX_PROPERTY_STRING = 64

const int Irisys.BlackfinAPI.Blackfin.MAX_DEVICEID_STRING = 160

8.1.5 Property documentation

int Irisys.BlackfinAPI.Blackfin.SerialNumber [get], [set]

The unique serial number of the Irisys IP device
GetSerialNumber()

int Irisys.BlackfinAPI.Blackfin.CountLogPeriod [get], [set]

The period at which count values are logged to the flash memory in the device
GetCountLogPeriod()

string Irisys.BlackfinAPI.Blackfin.MACAddress [get], [set]

The unique MAC address of the IP device
GetMACAddress()

IPAddress Irisys.BlackfinAPI.Blackfin.ClientConfigIP [get], [set]

The IP address the IP counter will attempt to connect to if ClientConfigEnable is set to true
GetClientConfigIP()

string Irisys.BlackfinAPI.Blackfin.ClientConfigHostname [get], [set]

The hostname the IP counter will attempt to connect to if ClientConfigEnable is set to true
GetClientConfigHostname()

ushort Irisys.BlackfinAPI.Blackfin.ClientConfigPort [get], [set]

The port number IP counter will attempt to connect to if ClientConfigEnable is set to true
GetClientConfigPortNumber()

ushort Irisys.BlackfinAPI.Blackfin.ClientConfigPortNumber [get]

uint Irisys.BlackfinAPI.Blackfin.ClientConfigTimeout [get], [set]

The time in seconds between reconnection attempts if ClientConfigEnable is set to true
GetClientConfigTimeout()

bool Irisys.BlackfinAPI.Blackfin.ClientConfigEnable [get], [set]

Whether or not the counter wil attempt to connect out to a server
GetClientConfigEnable()

List<Count> Irisys.BlackfinAPI.Blackfin.Counts [get], [set]

The Count Log containing historic count information
GetLastNCounts() GetCounts() GetCurrentCounts()

DeviceStatus Irisys.BlackfinAPI.Blackfin.DeviceStatus [get], [set]

The device diagnostic log
GetDeviceStatus()

DateTime Irisys.BlackfinAPI.Blackfin.UnitTime [get], [set]

The time on the device (UTC) as of the time when GetUnitTime() called.
GetUnitTime()

string Irisys.BlackfinAPI.Blackfin.LocaleString [get], [set]

A freeform user string
GetLocaleString()

string Irisys.BlackfinAPI.Blackfin.UserString [get], [set]

A freeform user string
GetUserString()

List<string> Irisys.BlackfinAPI.Blackfin.RegisterLabels [get], [set]

An ordered list of labels assigned to the enabled count registers on the device
GetRegisterLabels()

string Irisys.BlackfinAPI.Blackfin.DeviceName [get], [set]

A freeform user string
GetDeviceName()

string Irisys.BlackfinAPI.Blackfin.DeviceID [get], [set]

A freeform user string
GetDeviceID()

string Irisys.BlackfinAPI.Blackfin.SiteName [get], [set]

A freeform user string
GetSiteName()

string Irisys.BlackfinAPI.Blackfin.SiteID [get], [set]

A freeform user string
GetSiteID()

bool Irisys.BlackfinAPI.Blackfin.DHCPEnabled [get], [set]

Whether or not the device uses DHCP to obtain an IP address
GetDHCPEnabled()

IPAddress Irisys.BlackfinAPI.Blackfin.IPAddress [get], [set]

The static IP address for the counter to use if DHCPEnabled is false
GetIPAddress()

IPAddress Irisys.BlackfinAPI.Blackfin.SubnetMask [get], [set]

GetSubnetMask()

IPAddress Irisys.BlackfinAPI.Blackfin.Gateway [get], [set]

GetGateway()

string Irisys.BlackfinAPI.Blackfin.IPFirmwareVersion [get], [set]

The firmware version for the IP board
GetIPFirmwareVersion()

string Irisys.BlackfinAPI.Blackfin.MonitorFirmwareVersion [get], [set]

The firmware version for the DSP head
GetMonitorFirmwareVersion()

IPAddress Irisys.BlackfinAPI.Blackfin.DNS1 [get], [set]

The primary DNS server the device will use to resolve host names
GetDNS1()

IPAddress Irisys.BlackfinAPI.Blackfin.DNS2 [get], [set]

The secondary DNS server the device will use to resolve host names
GetDNS2()

IPAddress Irisys.BlackfinAPI.Blackfin.DNS3 [get], [set]

The tertiary DNS server the device will use to resolve host names
GetDNS3()

UInt64 Irisys.BlackfinAPI.Blackfin.UpTime [get], [set]

The amount of seconds since last reset
GetUpTime()

IrisysTimeZone Irisys.BlackfinAPI.Blackfin.UnitTimeZone [get], [set]

The time zone in which the device resides
GetUnitTimeZone()

String Irisys.BlackfinAPI.Blackfin.NetworkChecksum [get], [set]

The network settings checksum generated by the last successful call to GetNetworkChecksum()
This command was introduced in the 3.0.x series of APIs
GetNetworkChecksum()

int Irisys.BlackfinAPI.Blackfin.PacketTimeout [get]

The maximum time to wait for a response from the device before flagging a comms call as timed out

8.2 Irisys.BlackfinAPI.BlackfinEngine class reference

Public Types

- enum ConnectionErrorType { ConnectionErrorType.FATAL, ConnectionErrorType.WARNING }

Public Member Functions

- delegate void ConnectionErrorHandler (Blackfin sender, string errorString, Exception exp, ConnectionErrorType errorType)
- BlackfinEngine ()
- void StartEngine ()
- void ShutdownEngine ()
- bool AddNewCounterEndPoint (Blackfin counter, Socket socket)
- bool AddNewCounterEndPoint (Blackfin counter, Socket socket, ConnectionErrorHandler error)
- bool AddNewCounterEndPoint (Blackfin counter, System.IO.Ports.SerialPort serialPort, ConnectionErrorHandler error)
- bool AddNewCounterEndPoint (Blackfin counter, System.IO.Ports.SerialPort serialPort)
- object RemoveCounterEndPoint (Blackfin counter)
- void ClockCounters ()

Static Public Member Functions

- static string GetAPIVersion ()

8.2.1 Detailed description

Class for connecting Irisys.BlackfinAPI.Blackfin objects to a serial port or Socket

8.2.2 Member enumeration documentation

enum Irisys.BlackfinAPI.BlackfinEngine.ConnectionErrorType

Types of connection error possible

Enumerator

FATAL

WARNING

8.2.3 Constructor & destructor documentation

Irisys.BlackfinAPI.BlackfinEngine.BlackfinEngine ( )

Instantiates a new Blackfin Engine ready to be initialized.

8.2.4 Member function documentation

delegate void Irisys.BlackfinAPI.BlackfinEngine.ConnectionErrorHandler (Blackfin *sender,* string *errorString,* Exception *exp,* ConnectionErrorType *errorType*)

Error handler interface for Blackfin API This allows a client to recieve a call back when an error occurs

Parameters

sender	The Blackfin connection on which the error occured
errorString	A description of the error
exp	The exact exception thrown

errorType	Whether the error was fatal or a warning

static string Irisys.BlackfinAPI.BlackfinEngine.GetAPIVersion ( ) [static]

Returns a string representation of the version of the executing API

Returns

A string containing a version number

void Irisys.BlackfinAPI.BlackfinEngine.StartEngine ( )

Clocks the Blackfin object, which also will clock all virtual blackfins

void Irisys.BlackfinAPI.BlackfinEngine.ShutdownEngine ( )

Shuts down the Blackfin Engine and releases resources This stops network activity for any connected Blackfin devices, but does not disconnect them. You should call RemoveCounter-EndPoint for all Blackfin objects BEFORE calling this method.

bool Irisys.BlackfinAPI.BlackfinEngine.AddNewCounterEndPoint (Blackfin *counter,* Socket *socket*)

Connects a Blackfin object to a connected Socket object. This enabled the Blackfin to establish a connection and communicate with the remote device

Returns

True on connection success, False on failure

bool Irisys.BlackfinAPI.BlackfinEngine.AddNewCounterEndPoint (Blackfin *counter,* Socket *socket,* ConnectionErrorHandler *error*)

Connects a Blackfin object to a connected Socket object. This enabled the Blackfin to establish a connection and communicate with the remote device The error interface will be informed of all connection errors, so you should check that the counter is the relevant object

Parameters

counter	The Blackfin object to connect
socket	A connected socket to the device
error	An error callback interface.

Returns

True on connection success, False on failure

bool Irisys.BlackfinAPI.BlackfinEngine.AddNewCounterEndPoint (Blackfin *counter,* System.IO.Ports.SerialPort *serialPort,* ConnectionErrorHandler *error*)

Connects a Blackfin object to a connected SerialPort object. This enabled the Blackfin to establish a connection and communicate with the remote device The error interface will be informed of all connection errors, so you should check that the counter is the relevant object

Parameters

counter	
serialPort	
error	

Returns

True on connection success, False on failure

bool Irisys.BlackfinAPI.BlackfinEngine.AddNewCounterEndPoint (Blackfin *counter,* System.IO.Ports.SerialPort *serialPort*)

Connects a Blackfin object to a connected Socket object. This enabled the Blackfin to establish a connection and communicate with the remote device

Parameters

counter	The Blackfin object to connect
serialPort	The open serial port connected to the remote device

Returns

True on connection success, False on failure

object Irisys.BlackfinAPI.BlackfinEngine.RemoveCounterEndPoint (Blackfin *counter*)

Disconnects the Blackfin object from the remote device. The underlying port will not be closed, this is the responsibility of the calling code.

Parameters

counter	The Blackfin object to disconnect

Returns

Returns the connection object used (of type Socket or SerialPort) or null if device was not connected

void Irisys.BlackfinAPI.BlackfinEngine.ClockCounters ( )

This should be used if the user wants to clock the counters thenselves without an extra thread

8.3 Irisys.BlackfinAPI.Interfaces.Count class reference

Public Attributes

- List< uint > countLines = new List<uint>()
- DateTime countTime

8.3.1 Detailed description

A single count log entry

8.3.2 Member data documentation

List<uint> Irisys.BlackfinAPI.Interfaces.Count.countLines = new List<uint>()

A list of count line count values

DateTime Irisys.BlackfinAPI.Interfaces.Count.countTime

A timestamp for this count log entry

8.4 Irisys.BlackfinAPI.Interfaces.DeviceLogEntry class reference

Public Attributes

- string errorDescription = ""
- DateTime timestamp

8.4.1 Detailed description

A single device log entry.
This could be an error, warning or message

8.4.2 Member data documentation

string Irisys.BlackfinAPI.Interfaces.DeviceLogEntry.errorDescription = ""

A string describing the condition of the counter

DateTime Irisys.BlackfinAPI.Interfaces.DeviceLogEntry.timestamp

A timestamp when the entry was logged

8.5 Irisys.BlackfinAPI.Interfaces.DeviceStatus class reference

Public Attributes

- List< DeviceLogEntry > m_warnList = new List<DeviceLogEntry>()
- List< DeviceLogEntry > m_errorList = new List<DeviceLogEntry>()
- List< DeviceLogEntry > m_infoList = new List<DeviceLogEntry>()

8.5.1 Detailed description

A class representing the status of the device

8.5.2 Member data documentation

List<DeviceLogEntry> Irisys.BlackfinAPI.Interfaces.DeviceStatus.m_warnList = new List<DeviceLogEntry>()

A list of all the warnings in the device diagnostic log

List<DeviceLogEntry> Irisys.BlackfinAPI.Interfaces.DeviceStatus.m_errorList = new List<DeviceLogEntry>()

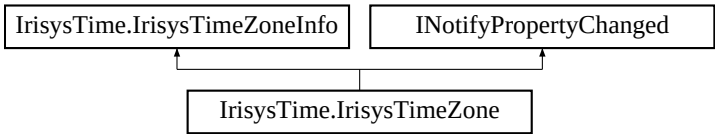
A list of all the errors in the device diagnostic log

List<DeviceLogEntry> Irisys.BlackfinAPI.Interfaces.DeviceStatus.m_infoList = new List<DeviceLogEntry>()

A list of all of the information messages in the device diagnostic log

8.6 IrisysTime.IrisysTimeZone class reference

Inheritance diagram for IrisysTime.IrisysTimeZone:



Public Member Functions

- IrisysTimeZone (IrisysTimeZoneInfo info)

Properties

- bool ApplyDST `[get, set]`

Events

- PropertyChangedEventHandler PropertyChanged

Additional Inherited Members

8.6.1 Detailed description

A mutable class representing a time zone which can be set in the flash table

8.6.2 Constructor & destructor documentation

IrisysTime.IrisysTimeZone.IrisysTimeZone (IrisysTimeZoneInfo *info*)

Parameters

info	

8.6.3 Property documentation

bool IrisysTime.IrisysTimeZone.ApplyDST `[get], [set]`

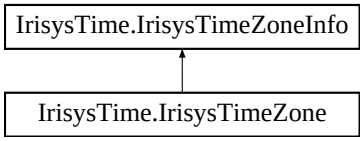
Flag showing whether or not Daylight Savings time should be applied. Note that this is a user setting and has no effect on the device.

8.6.4 Event documentation

PropertyChangedEventHandler IrisysTime.IrisysTimeZone.PropertyChanged

8.7 IrisysTime.IrisysTimeZoneInfo class reference

Inheritance diagram for IrisysTime.IrisysTimeZoneInfo:



Public Member Functions

- override bool Equals (object obj)
- override int GetHashCode ()
- override string ToString ()

Static Public Attributes

- static readonly IrisysTimeZoneInfo UtcTimeZone = timeZones[0]

Protected Member Functions

- IrisysTimeZoneInfo (TimeSpan t, bool b, string id)

Properties

- TimeSpan BaseUtcOffset `[get, set]`
- bool SupportsDST `[get, set]`
- string TimeZoneID `[get, set]`
- string DisplayName `[get]`
- static IEnumerable < IrisysTimeZoneInfo > TimeZones `[get]`

8.7.1 Detailed description

Immutable class representing an avaliable TimeZone for a Blackfin device

8.7.2 Constructor & destructor documentation

IrisysTime.IrisysTimeZoneInfo.IrisysTimeZoneInfo (TimeSpan *t,* bool *b,* string *id*) `[protected]`

Creates a new time zone with the specified parameters
Parameters

t	The base UTC offset
b	Whether or not DST is supported
id	The Windows Time zone ID

8.7.3 Member function documentation

override bool IrisysTime.IrisysTimeZoneInfo.Equals (object *obj*)

Compares timezones by the TimeZoneID.
Parameters

obj	

Returns

override int IrisysTime.IrisysTimeZoneInfo.GetHashCode ( )

Returns the TimeZoneID hashcode

Returns

override string IrisysTime.IrisysTimeZoneInfo.ToString ( )

Returns the three properties, semicolon delimited

Returns

8.7.4 Member data documentation

readonly IrisysTimeZoneInfo IrisysTime.IrisysTimeZoneInfo.UtcTimeZone = timeZones[0] `[static]`

The UTC Timezone

8.7.5 Property documentation

TimeSpan IrisysTime.IrisysTimeZoneInfo.BaseUtcOffset `[get], [set]`

The base UTC offset for the timezone

bool IrisysTime.IrisysTimeZoneInfo.SupportsDST `[get], [set]`

Whether or not the timezone can use DST (NOT whether it is using it)

string IrisysTime.IrisysTimeZoneInfo.TimeZoneID `[get], [set]`

The windows ID string of the timezone

string IrisysTime.IrisysTimeZoneInfo.DisplayName `[get]`

A string representation of the timezone

IEnumerable<IrisysTimeZoneInfo> IrisysTime.IrisysTimeZoneInfo.TimeZones `[static], [get]`

All of the Irisys supported time zones

8.8 Irisys.BlackfinAPI.SerialNumberConverter class reference

Static Public Member Functions

- static string ConvertSerialNumber (int serialNumber)

8.8.1 Detailed description

Utility class for conversions

8.8.2 Member function documentation

static string Irisys.BlackfinAPI.SerialNumberConverter.ConvertSerialNumber (int *serialNumber*) `[static]`

Converts a long serial number to an alpha numeric serial number, as displayed on the IRC3000 series hardware.

Parameters

serialNumber	A numeric serial number returned from the Blackfin.SerialNumber property

Returns

The alpha numeric serial as a string

InfraRed Integrated Systems Limited

Park Circle Tithe Barn Way
Swan Valley
Northampton NN4 9BG UK
Tel: **+44 (0) 1604 594 200**
Fax: **+44 (0) 1604 594 210**
Email: **support@irisys.co.uk**
sales@irisys.co.uk

Web site: **www.irisys.co.uk**

Irisys Americas

One Glenlake Parkway
Suite 700
Atlanta GA 30328 USA
Tel: **+1 678 638 6248**
Email: **support@irisys.net**
sales@irisys.net

Web site: **www.irisys.net**

July 2014
IPU 40302
Issue 4