

DS8R APPLICATION PROGRAMMING INTERFACE

Introduction

Contained within the DS8R Software Installation is an Application Programming Interface (API) allowing bespoke programs to control all connected DS8R devices.

The API takes the form of a Common Object Model (COM) out-of-process Server. Through a few exposed interfaces, the COM Server enables applications to manage one or more DS8R Devices connected to the PC. The COM Server manages all the raw communications between the PC and the connected devices.

To be able to make use of the API, the chosen programming language must be able to create references to Windows COM objects and support the IDispatch interface. Programming languages known to provide this support are, but not limited to, C++, Visual Basic, C#, Python & Delphi. Additionally, it is possible to create references to COM Objects in MatLab, LabView & Octave. Although these applications may not provide full support for features such as event management.

Implementation

When designing applications that employ the DS8R COM Server, it is important to appreciate that from the point of executing a change on one or more of the devices, it is not possible to guarantee the time taken before the device/devices have completed the update. Although PC are generally very quick, if it is necessary to synchronise software generated pulse triggers with external devices, then it is recommended that an external trigger source is used, connected to the trigger input on the rear of the unit.

Examples

Example code for Visual Basic, Python and MatLab can be found at the end of this document.

CONTENTS

IDS8RController Object.....	3
IDS8RController.GetState.....	4
IDS8RController.SetState.....	5
IDS8RStateCollection Object.....	6
IDS8RStateCollection.Count.....	7
IDS8RStateCollection.Items.....	8
IDS8RState Object.....	9
IDS8RState.Demand.....	10
IDS8RState.PulseWidth.....	11
IDS8RState.DemandLimit.....	12
IDS8RState.Recovery.....	13
IDS8RState.Dwell.....	14
IDS8RState.Mode.....	15
IDS8RState.Polarity.....	16
IDS8RState.Source.....	17
IDS8RState.Buzzer.....	18
IDS8RState.Trigger.....	19

IDS8RCONTROLLER OBJECT

Remarks

A **IDS8RController** object provides the fundamental functionality to enable obtaining and changing the state of DS8R devices connected to the computer, either individually or all.

On creating the first instance of the object, a inventory of all the connected devices is made at which point their state is obtained.

The object exposes two methods `GetState` and `SetState` through which it is possible to obtain the instantaneous state as well as modify the current state.

Inheritance

IDispatch
IUnknown

Methods

Methods	Description
GetState	Returns an <code>IDS8RStateCollection</code> object representing the instantaneous state of all the connected devices.
SetState	Takes either an <code>IDS8RStateCollection</code> or <code>IDS8RState</code> objects and applies the parameters to the connected devices.

Visual Basic

```
Dim DS8RServer as DS8RComServer.IDS8RController
Dim Devices as DS8RComServer.IDS8RStateCollection
Dim DeviceState as DS8RComServer.IDS8RState

DS8RServer = new DS8RComServer.IDS8RController

Devices = DS8RServer.GetState
If (Devices.Count > 0) Then
    DeviceState = Devices.Items(0)

    \... Modify parameters

    DS8RServer.SetState DeviceState, True

End If
```

IDS8RCONTROLLER.GETSTATE

Parameters None

Returns IDS8RStateCollection object

Description The returned object contains the instantaneous state of the system at the time the function was called.

The object lifetime will exist until all references to have been freed. This includes references to derived objects.

Will ALWAYS return an object, even if there are no devices connected.

Syntax	Visual Basic
<code>object.GetState</code>	

Part	Description
<code>object</code>	Required, always the name of a IDS8RController object.

IDS8RCONTROLLER.SETSTATE

Parameters IDS8RStateCollection object OR IDS8RState object

Blocking Boolean (Optional)

Returns Nothing

Description When passed either a IDS8RStateCollection object or a IDS8RState object which has had any of the device parameters modified, will cause the changes to be sent to either all connected DS8R devices or just the specific DS8R device.

An optional parameter Blocking allowing the caller to specify if the call returns AFTER the devices/device have been completely updated OR if the call returns immediately with the devices being update asynchronously.

Syntax	Visual Basic
<pre><i>object</i>.SetState state , blocking</pre>	

Part	Description
<i>object</i>	Required, always the name of a IDS8RController object.
<i>state</i>	Required, name of either an IDS8RStateCollection or IDS8RState object.
<i>blocking</i>	Optional, Boolean value that indicates if the call should block until the device has been fully updated.

IDS8RSTATECOLLECTION OBJECT

Remarks

A `IDS8RStateCollection` object represents the state of all DS8R devices at the time the object was obtained from `IDS8RController.GetState`.

Each device state is represented as a `IDS8RState` object which is obtained from the `Items` method.

Once a `IDS8RStateCollection` state is obtained, any changes made to the devices by another application will NOT be reflected in this object.

Inheritance

`IDispatch`

`IUnknown`

Methods

Methods	Description
<u>Count</u>	Returns the number of items in the collection.
<u>Items</u>	Provides access to an array of <code>IDS8RState</code> . Items are zero based.

Visual Basic

```
Dim DS8R as DS8RComServer.DS8RServer
Dim Devices as DS8RComServer.DS8RDevices
Dim State as DS8RComServer.DS8RDevice
Dim index as Integer

DS8R = new DS8RComServer.DS8RServer
Devices = DS8R.GetState

If (Devices.Count > 0) then
  For index = 0 to (Devices.Count -1 )
    State = Devices.Items( Index )
    State.Demand = 1000 '10 mA
  Next Index

  DS8R.SetState Devices, True
End If
```

IDS8RSTATECOLLECTION.COUNT

Parameters None

Returns Number of DS8RDevice objects items available in then collection.

Description Returns the number of items that **Items(index)** will return.

Syntax	Visual Basic
<i>object.Count</i>	

Parts	Description
<i>object</i>	the name of an object of type IDS8RStateCollection

IDS8RSTATECOLLECTION.ITEMS

Parameters Zero based index of item to retrieve.

Returns A IDS8RState object representing the device state at the time IDS8RController.GetState was called.

Remarks If the value of index is not in the range of 0 .. (Count – 1) then the returned object is **Nothing**.

Syntax	Visual Basic
<pre>object.Items (index)</pre>	

Parts	Description
<i>object</i>	Required, name of an IDS8RStateCollection object.
<i>index</i>	Zero based index into an array of IDS8RState objects.

IDS8RSTATE OBJECT

Remarks

IDS8RState object represents the state of a single DS8R device. The device can be identified using the **SerialNumber** property. The order of the devices within the **IDS8RStateCollection** collection is not guaranteed to be the same between sessions.

The device referenced by the object is not immediately updated when the properties are modified. It is necessary to call **IDS8RController.SetState** passing a **IDS8RStateCollection** collection object OR a **IDS8RState** object.

However, calling the **Trigger** method will execute the action immediately without the need to pass the object to **IDS8Server.SetState**.

Inheritance

IDSipatch

IUnknown

Methods

Methods	Description
Trigger	Will cause the referenced DS8R to immediately produce a stimulus pulse.

Properties

Property	Description
Demand	Stimulus pulse stimulus current in μA .
PulseWidth	Stimulus pulse width in μs .
DemandLimit	Maximum stimulus current as well as scaling for the input control voltage in μA .
Recovery	Recovery pulse percentage.
Dwell	Dwell period between the stimulus and recovery pulses in μs .
Mode	Pulse mode can be one of the following: DS8RModeMonoPhasic or DS8RModeBiphasic.
Polarity	Polarity of the stimulus pulse. One of the following: DS8RPolarityPos, DS8RPolarityNeg or DS8RPolarityAlt.
Source	Demand source selecting between front panel control and input control voltage. One of the following: DS8RSourceInternal or DS8RSourceExternal.
Buzzer	Out of Compliance warning enabled state. Boolean value.

IDS8RSTATE.DEMAND

Remarks Integer value representing the stimulus pulse current demand in micro-amps (μA) with a resolution of $100\mu\text{S}$

Valid Demand values must be in the range of 0 to 1A ($1000000\mu\text{A}$).

However, if a DemandLimit has been set, attempting to set the Demand to a value greater than this value will result in the Demand being set to the DemandLimit.

Syntax	Visual Basic
<pre><i>value</i> = <i>object</i>.Demand</pre>	
<pre><i>object</i>.Demand = <i>value</i></pre>	

Parts	Description
<i>object</i>	Required, name of a IDS8RState object.
<i>value</i>	Required, Integer value representing the Demand in μA .

IDS8RSTATE.PULSEWIDTH

Remarks Represents the stimulus pulse width in μs with a resolution of $1\mu\text{s}$.

Valid values must be in the range of $10\mu\text{s}$ up to $2000\mu\text{s}$.

Syntax	Visual Basic
<code>value = object.PulseWidth</code>	
<code>object.PulseWidth = value</code>	

Parts	Description
<code>object</code>	Required, name of a DS8RDevice object.
<code>value</code>	Required, Integer value representing the pulse width in μs .

IDS8RSTATE.DEMANDLIMIT

Remarks Integer value representing the maximum demand in μA with a resolution of 1mA.

If a demand limit is set to a value that is lower than the current demand value, the demand value will be set to the new demand limit.

Valid values must be in the range 100mA to 1000mA.

Syntax	Visual Basic
<pre><i>value</i> = <i>object</i>.DemandLimit</pre>	
<pre><i>object</i>.DemandLimit = <i>value</i></pre>	

Parts	Description
<i>object</i>	Required, name of a IDS8RState object.
<i>value</i>	Required, Integer value representing the Demand Limit in μA .

IDS8RSTATE.RECOVERY

Remarks Integer value of the stimulus pulse recover percentage. The value has a resolution of 1%.

Valid values must be in the range 10% to 100%.

Syntax	Visual Basic
<pre><i>value</i> = <i>object</i>.Recovery</pre>	
<pre><i>object</i>.Recovery = <i>value</i></pre>	

Parts	Description
<i>object</i>	Required, name of a IDS8RState object.
<i>value</i>	Required, integer value percentage.

IDS8RSTATE.DWELL

Remarks Integer value of the current delay between the end of the stimulus pulse and the start of the recovery pulse. The value give is in whole micro-seconds (μs).

Valid values are in the range 1us to 99us.

Syntax	Visual Basic
<pre>value = object.Dwell</pre>	
<pre>object.Dwell = value</pre>	

Parts	Description
<i>object</i>	Required, name of a IDS8RState object.
<i>value</i>	<i>Required</i> , integer value dwell time in μs .

IDS8RSTATE.MODE

Remarks DS8RMode enumeration used to represent the stimulus pulse mode.

Valid values:

DS8RModeBiphasic
DS8RModeMono.

Syntax	Visual Basic
<code>value = object.Mode</code>	
<code>object.Mode = value</code>	

Parts	Description
<i>Object</i>	<i>Required, name of a DS8RDevice object.</i>
<i>value</i>	Required, value from DS8RMode enumeration.

IDS8RSTATE.POLARITY

Remarks DS8RPolarity enumeration which provides feedback on the stimulus polarity.

Valid values will be:

DS8RPolarityPos
DS8RPolarityNeg
DS8RPolarityAlt

Syntax	Visual Basic
<pre><i>value</i> = <i>object</i>.Polarity</pre>	
<pre><i>object</i>.Polarity = <i>value</i></pre>	

Parts	Description
<i>object</i>	Required, name of IDS8RState object
<i>value</i>	Required, value from DS8RPolarity enumeration.

IDS8RSTATE.SOURCE

Remarks DS8RSource enumeration detailing the demand controlling source.

Valid values are:

DS8RSourceInternal
DS8RSourceExternal.

Syntax	Visual Basic
<pre><i>value</i> = <i>object</i>.Source</pre>	
<pre><i>object</i>.Source = <i>value</i></pre>	

Parts	Description
<i>object</i>	Required, name of IDS8RState object.
<i>value</i>	Required, value from DS8RSource enumeration.

IDS8RSTATE.BUZZER

Remarks Boolean value representing the state of the Out of Compliance audible warning. When True, the buzzer is enabled.

Syntax	Visual Basic
<pre><i>value</i> = <i>object</i>.Buzzer</pre>	
<pre><i>object</i>.Buzzer = <i>value</i></pre>	

Parts	Description
<i>object</i>	Required, name of a DS8RDevice object.
<i>value</i>	Required, Boolean value of the state of the Out of Compliance buzzer.

IDS8RSTATE.TRIGGER

Remarks triggers a single stimulus pulse.

This function is performed immediately without the need to pass the object to `IDS8RController.SetState`.

If accurate timing of the stimulus pulse is required, it is recommended to use an external trigger source connected to the Trigger input connector.

Syntax	Visual Basic
<code><i>object</i>.Trigger</code>	

Parts	Description
<code><i>object</i></code>	Required, name of a <code>IDS8RState</code> object.

VISUAL BASIC EXAMPLE

Visual Basic**Example**

```
Dim DS8R as DS8R.DS8RController

Private Sub ReleaseControllerReference ()
    If Not (DS8R is Nothing) then
        DS8R = Nothing
        GC.Collect()
    End If
End Sub

Private Sub CreateControllerReference()
    If (DS8R is Nothing) then
        DS8R = New DS8R.DS8RController
    End If
End Sub

Private Sub ReadFirstDevice()
    Dim Collection as DS8R.DS8RStateCollection
    Dim State as DS8R.DS8RState

    CreateControllerReference()
    Collection = DS8R.GetState

    If (Collection.Count > 0) Then
        State = Collection.Items(0)
        edDemand.Text = State.Demand
        edPulseWidth.Text = State.PulseWidth
        edRecovery.Text = State.Recovery
        edDwell.Text = State.Dwell
    End If
End Sub

Private Sub WriteFirstDevice()
    Dim Collection as DS8R.DS8RStateCollection
    Dim State as DS8R.DS8RState

    CreateControllerReference()
    Collection = DS8R.GetState

    If (Collection.Count > 0) Then
        State = Collection.Items(0)
        State.Demand = edDemand.Text
        State.PulseWidth = edPulseWidth.Text
        State.Recovery = edRecovery.Text
        State.Dwell = edDwell.Text

        DS8R.SetState(State)
    End If
End Sub
```

PYTHON EXAMPLE

The following example requires the “pywin32” module to be installed.

Python	Example
<pre>import win32com.client import tkinter as tk from tkinter import ttk global DS8R def CreateDS8RReference(): global DS8R DS8R = win32com.client.Dispatch("DS8R.DS8RController") def ChangeDemand(NewDemand): global DS8R Collection = DS8R.getState if (Collection.Count > 0): State = collection.Items(0) State.Demand = NewDemand DS8R.SetState(state) def Button1_Click(): ChangeDemand(10000) def Button2_Click(): ChangeDemand(25000) def main(): global DS8R CreateDS8RReference() window = tk.Tk() window.geometry("350x350") window.title("Digitimer DS8R Python Example") Button1 = ttk.Button(window, text="Set 10mA", command=Button1_Click) Button1.pack() Button2 = ttk.Button(window, text="Set 25mA", command=Button2_Click) Button2.pack() window.mainloop() DS8R = None</pre>	

MATLAB/OCTAVE EXAMPLE

The following example has only been tested on Octave. However, it is believed that this should also be true for MatLab.

MatLab/Octave

Example

```
clear;
clc;
clf;

pkg load windows;
global ds8r;

function devCount = WaitForDevice
    global ds8r;
    ds8r = actxserver("DS8R.DS8RController");
    nSeconds = time() + 5;
    devCount = 0;
    while ((time() < nSeconds) && (devCount == 0))
        collection = ds8r.GetState;
        devCount = collection.Count;
    endwhile
endfunction

function ShowDeviceState(State)
    printf("Serial Number = %d\n", State.SerialNumber);
    printf("Demand = %d uA\n", State.Demand);
    printf("DemandLimit = %d uA\n", State.DemandLimit);
    printf("Pulse Width = %d us\n", State.PulseWidth);
    printf("Recovery = %d %%\n", State.Recovery);
    printf("Dwell = %d us\n", State.Dwell);
endfunction

function ToggleDeviceDemand(State)
    global ds8r;
    if (State.Demand==0)
        State.Demand = 10000;
    else
        State.Demand = 0;
    endif
    ds8r.SetState(State);
endfunction

devCount = WaitForDevice();
if (devCount>0)
    collection = ds8r.GetState;
    state = collection.items(0);
    ShowDeviceState(state);
    ToggleDeviceDemand(state);
endif

printf("\nFinished\n");
```