



## ***Using Data Translation® DT8837 Instrument Modules with MATLAB® through IVI-COM***

RTF-24191-A

### ***Introduction***

You can access the functionality of Data Translation's DT8837 instrument modules using the MATLAB Instrument Control Toolbox. The Instrument Control Toolbox communicates with DT8837 instrument modules through a MATLAB driver (wrapper) to the DT8837 IVI-COM driver.

To use the MATLAB Instrument Control Toolbox with a DT8837 instrument module, you need to install the DT8837 IVI-COM driver provided by Data Translation. The DT8837 IVI-COM driver setup program also installs the MATLAB driver for the DT8837 (DT8837\_DT8837.mdd) and example programs that illustrates how to use MATLAB with the IVI-COM driver.

This document describes how to access the functionality of a DT8837 instrument module through the MATLAB Instrument Control Toolbox.

### ***Prerequisite Software***

You must first install VISA (Virtual Instrument Software Architecture) software before installing the DT8837 IVI-COM driver.

To install VISA, do the following:

1. Go to **www.agilent.com**, enter **IO Libraries Suite** in the search field, and select **Agilent IO Libraries Suite 15.0** from the search results.
2. Follow the instructions on Agilent's web site to download and install the Agilent IO Libraries, which include VISA support, VISA COM support, and the Agilent Connection Expert tool.

To use the MATLAB Driver for the DT8837, ensure that you have installed version R2009b or higher of MATLAB and the Instrument Control Toolbox.

# ***Installing the DT8837 IVI-COM Driver and the MATLAB Driver for the DT8837***

To install the DT8837 IVI-COM driver and MATLAB driver for DT8837, perform the following steps:

1. Insert the DT8837 CD (supplied with your DT8837 instrument module) into your CD-ROM or DVD drive.  
*The installation program should automatically start, and the DT8837 installation program should appear.*
2. If the installation program does not automatically start, double-click **Setup.exe** from the CD.  
*The DT8837 installation program appears.*
3. Click **Install from Web (recommended)** to get the latest version of the software or **Install from CD** to install the software from the CD.
4. If you are installing from the web, click **DT8837 Software** to install the DT8837 software (including the DT8837 Getting Started Application, IVI-COM driver, IVI shared components, Eureka Discovery Utility, DT8837 SCPI Support, DT8837 Calibration Utility) and related documentation.
5. If you are installing from the DT8837 CD, click **Install Features**. When you are finished with the DT8837 CD, click **Quit Installer**.

---

**Note:** The MATLAB driver for DT8837 (DT8837\_DT8837.mdd) is available for download from the MATLAB Central website ([www.mathworks.com](http://www.mathworks.com)).

---

# Using the DT8837 IVI-COM Driver in MATLAB

Once you have installed the DT8837 IVI-COM driver and the MATLAB driver for DT8837, start MATLAB and ensure that the DT8837\_DT8837.mdd file is in the current directory window. If this file is not in the current directory window, browse to the directory in which you downloaded the DT8837\_DT8837.mdd file, and save this file to your working directory. Alternatively, you can change the MATLAB search path. The Path Browser allows you to add directories that MATLAB searches through. Refer to your MATLAB documentation for more information.

You can access DT8837 instrument modules from the MATLAB command line using IVI-COM methods and properties. For more information on IVI concepts, you can access the IVI-COM Driver Reference. From the Windows Start menu, click **Programs -> IVI -> DT8837 -> Documentation**. The following sections describe typical operations when writing a MATLAB script.

---

**Note:** The MATLAB Instrument Control Toolbox provides the graphical Test & Measurement Tool that allows you to access DT8837 instrument modules without writing MATLAB scripts; the tool generates the MATLAB script for you. See [www.mathworks.com/tmtool](http://www.mathworks.com/tmtool) for more information.

---

## Create an Object for the DT8837

Create an object for the DT8837 instrument module using the following command:

```
>> devObj = icdevice('DT8837_DT8837', devRsrcName);
```

where, *RsrcName* is an IVI logical name or an instrument-specific string, such as a VISA resource descriptor, that identifies the address of the DT8837 instrument module, and *dev* is the object that is created.

Specify `TCPIP::Address::INSTR` for *RsrcName*, where *Address* is the IP address of the instrument on the network. An example follows:

```
>> dev = icdevice('DT8837_DT8837',  
    'TCPIP::192.43.218.69::INSTR'; or  
    'TCPIP::192.43.218.69::SOCKET' *  
* use of SOCKET does not require VISA I/O Library
```

You can determine the IP address of your instrument on the TCP/IP network using an LXI discovery tool, such as the Data Translation Eureka Discovery Utility, automatically installed with the DT8837 CD. To use the Eureka Discovery Utility, perform the following steps: From the Windows Start menu, click **Programs -> Data Translation, Inc -> Instruments -> Eureka LXI Instrument Discovery**.

## Examples

There are three example programs available for download with the DT8837 MATLAB driver:

1. This example demonstrates how to synchronize the trigger on two devices using the trigger bus and perform continuous analog input on both devices.
2. This example demonstrates how to perform analog input, tachometer, and counter measurements.
3. This example demonstrates how to write a value to the digital output port and verify the value written by reading back the value from the same port.

The following example shows how to access a DT8837 instrument from the MATLAB Instrument Control Toolbox using the DT8837 IVI-COM driver:

```
%% Analog input, Tachometer, and Counter Measurements
%
% Copyright (C) 2010 DataTranslation Inc.
%

%% Introduction
% This example demonstrates how to perform analog input,
% tachometer, and counter measurements.

%%
% Open connection to instrument.
% devRsrcName is an IVI logical name or an instrument specific string
% that identifies the address of the instrument, such as a VISA
% resource descriptor string.
devRsrcName = 'TCPIP::192.43.218.135::INSTR';

% Create device object
deviceObj = icdevice('DT8837_DT8837', devRsrcName);

try
    %% Connect device object to the DT8837 instrument
    connect(deviceObj);

    %% Get instrument identity

    comobj = get(deviceObj, 'Identity');
    propertyValue = get(comobj, 'InstrumentModel');
    str = strcat ('InstrumentModel= ',propertyValue);
    disp(str);
    propertyValue = get(comobj, 'InstrumentManufacturer');
    str = strcat ('InstrumentManufacturer= ',propertyValue);
    disp(str);
    propertyValue = get(comobj, 'InstrumentFirmwareRevision');
    str = strcat ('InstrumentFirmwareRevision= ',propertyValue);
```

```

disp(str);
propertyValue = get(comobj, 'Description');
str = strcat ('Description= ',propertyValue);
disp(str);
propertyValue = get(comobj, 'Identifier');
str = strcat ('Identifier= ',propertyValue);
disp(str);
propertyValue = get(comobj, 'Vendor');
str = strcat ('Vendor= ',propertyValue);
disp(str);
propertyValue = get(comobj, 'Revision');
str = strcat ('Revision= ',propertyValue);
disp(str);

%% Display the AnalogInputChannels collection
analogInputChannelCount = deviceObj.Analoginputchannels.Count;

% Preallocate array.
analogInputChannelNames={analogInputChannelCount};

for iLoop= 1:analogInputChannelCount
    analogInputChannelNames{iLoop, 1} =
        invoke(deviceObj.Analoginputchannels, 'Name', iLoop);
end
disp(analogInputChannelNames)

%% Enable the channels to be measured by the DT8837 instrument
% Note: The counter channels and Tachometer channel are enabled
% within the analog input channels collection and their data is
% embedded in the analog input stream.

groupObj = get(deviceObj, 'Analoginputchannels');
groupObj = groupObj(1);

% Enable analog input channel 1 measurement
AiChanObject = invoke(groupObj, 'Item', 'AD.1');
AiChanObject.Enabled = true;

% Enable Counter 1 measurement
groupObj = get(deviceObj, 'Analoginputchannels');
groupObj = groupObj(1);
Ctr1Object = invoke(groupObj, 'Item', 'Ctr.1');
Ctr1Object.Enabled = true;

% Enable Tachometer measurement
TachObject = invoke(groupObj, 'Item', 'Tach.1');
TachObject.Enabled = true;

```

```

%% Configure the analog input subsystem

GroupObj = get(deviceObj, 'Analoginputacquisition');
analogInputAcquisitionObj = GroupObj(1);

% Disable wrapping, so that data does not get overwritten in the
% hardware FIFO when the FIFO is full
set(analogInputAcquisitionObj, 'WrapEnabled', 'off');

% Set the clock frequency
set(deviceObj.Analoginputacquisition(1), 'SampleRate', 1000);

% Set the trigger source to trigger the instrument immediately
set(deviceObj.Analoginputtrigger(1), 'Source', 'IMMEDIATE');

%% Configure Counter 1

set(deviceObj.Counter(1), 'StartEdge',
    'DT8837CounterEdgeGateRising');
set(deviceObj.Counter(1), 'StopEdge',
    'DT8837CounterEdgeGateFalling');

% Disable the SelfClear flag. When disabled, the previous
% measurement value from the counter is held and returned on any
% read of the counter before the next continuous measurement
% operation is completed.
set(deviceObj.Counter(1), 'SelfClearEnable', 'off');

%% Configure the tachometer

set(deviceObj.Tach(1), 'TachEdgeType',
    'DT8837TachEdgeTypeRising');

% Enable the SelfClear flag. When enabled, clears the latched
% measurement value from the tachometer after it is read; zeros
% are returned on any subsequent reads of the tachometer before
% the next continuous measurement operation is completed.
set(deviceObj.Tach(1), 'SelfClearEnabled', 'on');

% Enable the use of the MSB of the value to indicate new or old
% data. If staleValueEnabled is 'on', the MSB of the value
% indicates whether the measurement is fresh or stale. When
% StaleValueEnabled is 'on', the MSB of the value is set
% to 0 to indicate fresh data or 1 to indicate stale data.
% Reading the value before the measurement is complete returns an
% MSB of 1 to indicate stale data.
set(deviceObj.Tach(1), 'StaleValueEnabled', 'on');

%% Initiate the measurement and get the data.

```

```

invoke(analogInputAcquisitionObj, 'Arm');
invoke(analogInputAcquisitionObj, 'Initiate');

isRunning = false;
ScanIndex = 0;
RequestedScansToRead = 100;

% Wait for the analog input subsystem to change state to Running
% and the requested number of scans are read
while (isRunning == false || ScanIndex < (RequestedScansToRead-1))
    [ScanIndex, isRunning, isArmed, isTriggered, isADSyncDetected,
     isADFifoOverflow] = invoke(analogInputAcquisitionObj,
     'GetStatus', 0, 0, 0, 0, 0, 0);
end

% Read 100 scans from the instrument
RequestedScansIndex = 0;
[ActualScansIndex, ActualScansRead, StartTimeInSeconds,
 StartTimeInMilliseconds, samples] =
    invoke(analogInputAcquisitionObj, 'Fetch',
    int32(RequestedScansIndex), int32(RequestedScansToRead),
    int32(0), int32(0), int32(0), int32(0), [0;0]);

RequestedScansIndex = ActualScansIndex+ActualScansRead;
disp(['Actual Scans Index: ', num2str(ActualScansIndex) ,
     ' Actual Scans Read: ', num2str(ActualScansRead)]);

% reshape the samples into 3 rows (each row is associated with the
% channel where the first row is associated with analog input
% channel 1, the second row is associated with the tachometer
% channel and the third row is associated with counter 1 channel
B = reshape(samples,[3 100]);

% display the data for each channel
disp(B(1,:))
disp(B(2,:))
disp(B(3,:))

%% Stop the acquisition
invoke(analogInputAcquisitionObj, 'Abort');

catch DT8837error
    disp(['Error id: ', DT8837error.identifier]);
    disp(['Error Message: ',DT8837error.message]);
end

```

```
% Disconnect the device object from the instrument and remove it from  
% memory  
disconnect(deviceObj);  
delete(deviceObj);
```

## ***Conclusion***

To access DT8837 instruments from MATLAB, you need the MATLAB Instrument Control Toolbox, DT8837 IVI-COM driver, and MATLAB driver for DT8837. By combining the precise measurement capability of Data Translation with the powerful analytical capability of MATLAB, developers can create robust applications that solve difficult temperature measurement problems with ease.

## ***For More Information***

1. Overview of DT8837 Instrument Module:  
[www.datatranslation.com/products/dataacquisition/ethernet/dt8837.asp](http://www.datatranslation.com/products/dataacquisition/ethernet/dt8837.asp)
2. Overview of MATLAB software:  
[www.mathworks.com/matlab](http://www.mathworks.com/matlab)
3. Overview of MATLAB Instrument Control Toolbox:  
[www.mathworks.com/products/instrument](http://www.mathworks.com/products/instrument)



MATLAB® is a registered trademark of The MathWorks, Inc.

