med associates inc

instrumentation and software for research

DAQ2MPC

SOF-732-5

USER'S MANUAL

DOC-300
Rev. 1.1

Med Associates, Inc.
P.O. Box 319
St. Albans, Vermont 05478

Phone: 802.527.2343
Fax: 802.527.5095
www.med-associates.com

Table of Contents

Notes

Diagrams

CHAPTER 1 | GETTING STARTED

Introduction

DAQ2MPC enables Med-PC® users to collect data via Med-PC® protocols and Med Associates, Inc. hardware while simultaneously collecting data from other devices using a variety of analog input/output National Instruments (NI) data acquisition (DAQ) cards. To determine if a NI DAQ would be appropriate to use please visit this page: NI Supported Devices[1]

The function calls in the DAQ2MPC package allow the user to initialize and start and stop the DAQ card as well as retrieve data in user defined data sets. The DAQ card may be configured to sample data at a rate of 1Hz to 1KHz and has up to 5,000 data points available in the buffer.

Software Installation and Setup

Before beginning the software installation, phone, fax or e-mail Med Associates with the registration information.

Insert the **SOF-732-5 DAQ2MPC** CD into the CD drive and at the welcome screen click **Install**. If the CD does not auto-start, navigate to the CD drive in Windows Explorer and double click “autorun.exe” and click **Install** at the welcome screen.

1: http://digital.ni.com/public.nsf/allkb/B4E831774F29FB038625754C0081C050

CHAPTER 2 | DAQ2MPC FUNCTION CALLS

DAQConfigure

Syntax	`DAQConfigure(MG: MPCGlobalPtr, maxChannels: Real, samplesPerSecond: Real, terminalConfiguration: Real);`
Parameters	**MG**: The MED-PC Global Parameter. Used for passing error messages back to MED-PC. **maxChannels**: The number of channels to read data from. Default = 8. **samplesPerSecond**: The number of data points to read per second. Default = 100 Hz (samples per second). Available sample rate is 1 Hz – 1 KHz. **terminalConfiguration**: Referenced Single Ended = 1 (Default) Non-Referenced Single Ended = 2 Differential = 3 Pseudo-Differential = 4
Comments	The DAQConfigure function sets the Data Acquisition (DAQ) device's settings. Use this function in the InitializationCode section of the MedState Notation (MSN) program to ensure it runs before the first clock tick of the program's execution. The DAQConfigure function is optional, and is only required if the DAQ's default settings are unacceptable. Please refer to the DAQ device's documentation for information regarding terminal configuration.
Example	`InitializationCode = ~DAQConfigure(MG, 8, 100, 1);~` Initializes the DAQ device to read 8 channels, at 100Hz using referenced single ended inputs.

DAQIsRunning

Syntax	`DAQIsRunning(MG: MPCGlobalPtr): Real;`
Parameters	**MG**: The MED-PC Global Parameter.
Comments	Returns **TRUE** (1) if the DAQ functionality has fully initialized and is ready to start. Returns **FALSE** (0) if the DAQ is not ready to begin acquisition.
Example	`~B := DAQIsRunning(MG);~` Sets the variable B to the Boolean value returned by DAQIsRunning.

DAQStart

Syntax	`DAQStart(MG: MPCGlobalPtr);`
Parameters	**MG**: The MED-PC Global Parameter.
Comments	Starts reading data from the DAQ device. If the DAQ is already started, calling this procedure again has no effect.
Example	`~DAQStart(MG);~`

DAQStop

Syntax	`DAQStop(MG: MPCGlobalPtr);`
Parameters	**MG**: The MED-PC Global Parameter.
Comments	This procedure is optional and is not required. The procedure stops data collection for all boxes. *NOTE: Do not put this function in a program that normally runs with other programs at the same time. If the use of this function is desired then a separate MPC program should be written and this program should be run after all other programs have finished. If the DAQ is already stopped, calling this procedure again has no affect.*
Example	`~DAQStop(MG);~`

DAQReadChannel

Syntax	`DAQReadChannel(MG: MPCGlobalPtr;  channel: Real): Real;`
Parameters	**MG**: The MED-PC Global Parameter. **channel**: The channel to read the data point from. BOX is frequently used for this parameter because it allows the same program to be used in multiple boxes.
Comments	Returns the last datum point from the specified DAQ channel.
Example	`~A := DAQReadChannel(MG, BOX);~` Sets variable A's value to the last datum point read in the box running the MSN protocol. See **Using BOX As The Channel Number** for more information.

DAQClearBuffer

Syntax	`DAQClearBuffer(MG: MPCGlobalPtr;  channel: Real);`
Parameters	**MG**: The MED-PC Global Parameter. **channel**: The channel whose buffer is to be cleared. BOX is frequently used for this parameter because it allows the same program to be used in multiple boxes.
Comments	Clears the specified channel's data buffer that is used by **DAQReadDataPoints** and **DAQReadSequential**.
Example	`~DAQClearBuffer(MG, BOX);~` See **Using BOX As The Channel Number** for more information.

DAQReadDataPoints

Syntax	`DAQReadDataPoints(MG: MPCGlobalPtr; channel: Real; numDataPoints: Real;  X Var): Real;`
Parameters	**MG:** The MED-PC Global Parameter. **channel**: The channel from which to read data. BOX is frequently used for this parameter because it allows the same program to be used in multiple Boxes. **numDataPoints**: The number of data points to read from the channel. **X**: MSN array that receives the data. The array must be dimensioned with a value that is greater than, or equal to, numDataPoints. The array does not have to be named "X".
Comments	Copies **numDataPoints** from the buffer on the specified **channel** to array **X**. Returns 1 if **numDataPoints** were available in the buffer and copied to array **X**. Returns 0 (zero) if there were less than **numDataPoints** available in the buffer. Element 0 in the array is the oldest data point. The maximum number of data points available in the buffer is 5,000. **WARNING:** DAQReadDataPoints will copy duplicate values into the array if the function is called before **numDataPoints** of new values are available in the buffer.
Example	`~B := DAQReadDataPoints(MG, BOX, 200, C);~` Sets variable B to 1 if 200 data values were available and copied to array C, or sets B to 0 if less than 200 data values were available and therefore no data was copied to array C. See **Using BOX As The Channel Number** for more information.

DAQReadSequential

Syntax	`DAQReadSequential(MG: MPCGlobalPtr; channel: Real; numDataPoints: Real;  X Var): Real;`
Parameters	**MG**: The MED-PC Global Parameter. **channel**: The channel whose data is to be read. BOX is frequently used for this parameter because it allows the same program to be used in multiple boxes. **numDataPoints**: The number of data points to be read from the channel. **X**: MSN array that receives the data. The array must be dimensioned with a value that is greater than, or equal to, numDataPoints. The array does not have to be named "X".
Comments	Copies **numDataPoints** from the buffer on the specified **channel** to array **X** as long as all points in the set were not previously copied to the array. Returns 1 if **numDataPoints** were available in the buffer on the desired **channel** and copied to array **X**. Returns 0 (zero) if there were less than **numDataPoints** of new data values are available in the buffer. Element 0 in the array is the oldest data point. The maximum number of data points available in the buffer is 5,000.
Example	`~B := DAQReadSequential(MG, BOX, 200, Y);~` Sets variable B to 1 if 200 data values were available and copied to array Y, or sets B to 0 if less than 200 new data values were available and therefore no data was copied to array Y. See **Using BOX As The Channel Number** for more information.

CopyData

Syntax	`CopyData(MG: MPCGlobalPtr; numDataPoints: Real; Dest Var; Source Var);`
Parameters	**MG**: The MED-PC Global Parameter. **numDataPoints:** The number of data points to be copied. **Dest:** Reference to the start of memory block to receive copied data. **Source:** Reference to the source of data to be copied.
Comments	Copies data from one array into another array.

Example	`~CopyData (MG, 200, Z[Trunc(I))], Y);~`
	Copies 200 data points starting from the beginning of the Y array to the Z array, starting at element I. Variable I in MED-PC® is an Extended type, Trunc() must be used to convert it to an Integer type.
	`~CopyData (MG, 40, Q[80], W[120]);~`
	Copies 40 data points from the W array starting at element 121 (array indices are zero-based) to the Q array starting at element 81.

Using BOX As The Channel Number

When a program is loaded into a chamber (box) in MED-PC®, the chamber number that the program was loaded into is automatically stored and tracked by MED-PC® in that specific chamber's BOX variable. Therefore, when a program is loaded into chamber 1, chamber 1's BOX variable is assigned the value 1, when a program is loaded into chamber 2, chamber 2's BOX variable is assigned the value 2 etc.

Now, assume DAQ channel 1 is to be used for chamber 1, DAQ channel 2 is to be used for chamber 2 and so on up to chamber 8. When the program is loaded into the 8 chambers, each chamber's BOX variable will represent its chamber number, which in this example is the same as the DAQ channel being used for that chamber, so BOX can be used to indicate the DAQ channel in the function calls. This approach allows the same program to be used for all chambers instead of writing separate, chamber number specific programs for each chamber.

A similar approach could be used when multiple DAQ channels are needed for each chamber. Assume there are 4 chambers, and each chamber needs 2 DAQ channels. Channels 1-4 would be assigned sequentially to chambers 1-4, then channels 5-8 would be assigned to chambers 1-4 sequentially. To read channels 1-4 simply use BOX in the function call. To read channels 5-8 use BOX+4 in the function call. This approach again allows the same program to be used in all chambers.

CHAPTER 3 | SAMPLE CODE

```
\ List Data Variables Here
\  A   = The Most Recent Data Point from the Hardware
\  C() = When the 1st K1 Pulse was issued this was the
\        next 200 Data points read from the hardware.
\  D() = When the 2nd K1 Pulse was issued this was the
\        last 200 Data points read from the hardware.
\  Z() = All of the Data that has been read from the
\        hardware.  In this example up to 900,000 Data
\        points can be read before the array will be
\        full

\ List Working Variables Here
\  B   = Variable Used for Checking if the Data is Ready
\  I   = Subscript for Data Array Z
\  J   = Number of 200 Data Point Sets that have been Received
\  Y() = Temporary Array Used to get the Data from
\        the Hardware

DIM C = 199  \ Elements 0-199
DIM D = 199
DIM Y = 199
DIM Z = 900000

DISKVARS = A, C, D, Z

\ This code will be run before the first clock tick of an MSN
\ program's execution.  It is meant to ensure that certain
\ memory or data gets initialzied before the program runs.
\
InitializationCode = ~DAQConfigure(MG, 8, 100, 1);~
                     \ 8   = Number of Channels to read from
                     \ 100 = Samples Per Second (10ms Resolution)
                     \ 1   = Referenced Single Ended
\***************************************************
\      The code in this State Set is what most
\     customers will use with starting the data
\     reading and getting the most recent data
\             point from the hardware.
\***************************************************
S.S.1,
S1,     \ Wait for the Software Interface to be fully Initialized
```

```
  0.01": ~B := DAQIsRunning(MG);~;
         IF B = 1 [@Running, @Wait]
            @Running: SHOW 1,Awaiting Start Command,0 ---> S2
            @Wait:    ---> SX

S2,     \ Start the National Instruments Device Reading Data
  #START: ~DAQStart(MG);~; Z1 ---> S3

S3,     \ Display the most Recent Data Point from the Hardware
  0.01": ~A := DAQReadChannel(MG, BOX);~;
         SHOW 1,Load Cell Value,A ---> SX



\***************************************************
\  The code in this State Set shows how to log all
\      of the data readings from the hardware.
\***************************************************
S.S.2,
S1,     \ Seal the Array
  0.01": SET Z(I) = -987.987 ---> S2

S2,     \ Wait for Signal that the DAQ has started reading
  #Z1: ---> S3

S3,     \ Wait until the next 200 data points are ready
        \ With a 100 Samples Per Second (10ms Resolution)
        \ it will take 2s for 200 data points to be ready
  #K1:   ---> S5
  1.99": ---> S4

S4,     \ Get the Most recent 200 data points that have
        \ not already been read and then copy them into
        \ the Z Array for saving to the data file
  #K1: ---> S5
  0.01": ~B := DAQReadSequential(MG, BOX, 200, Y);~;
         IF B = 1 [@GotData, @Wait]
            @GotData: ADD J; SHOW 2,200 pts Rcvd,J;
                      ~CopyData(MG, 200, Z[Trunc(I)], Y);~;
                      SET I = I + 200;
                      SET Z(I) = -987.987;
                      IF I >= 900000 [@Max, @Cont]
                         @Max:  ---> S2
                         @Cont: ---> S3
            @Wait: ---> SX

S5,
  1": ---> SX


\***************************************************
\  The code in this State Set shows how to request
\    the next 200 data points from the hardware.
\***************************************************
S.S.3,
S1,     \ Wait for Signal that the DAQ has started reading
  #Z1: ---> S2

S2,     \ Clear the Buffer so that we know we are
        \ getting the next 200 measurements
        \
        \ NOTE: Calling this function will also affect
        \       the DAQReadSequential function in S.S.2
  #K1: ~DAQClearBuffer(MG, BOX);~ ---> S3

S3,     \ Wait until the next 200 data points are ready
        \ With a 100 Samples Per Second (10ms Resolution)
        \ it will take 2s for 200 data points to be ready
  1.99": ---> S4

S4,     \ Read the next 200 measurements
  0.01": ~B := DAQReadDataPoints(MG, BOX, 200, C);~;
         IF B = 1 [@GotData, @Wait]
            @GotData: SHOW 3,Next 200 pts Read,1 ---> S5
            @Wait: ---> SX
```

CHAPTER 4 | CONTACT INFORMATION

Please contact Med Associates, Inc. for information regarding any of our products.

Visit our website at www.med-associates.com for contact information.

For technical questions, call 802-527-2343 or email support@med-associates.com.