Clampex Application Note

How to Control Clampex from Another Application

Last revision: May 31, 2004

Clampex can be controlled from another application with a special API. This Application Note provides some basic guidance in how to do this.

Note that we cannot offer technical support on this matter, and the API may change without warning.

Use this document in conjunction with the header file AxClampexMsg.h and example VC++ code in the CLXMSG_TestBed project.

Before calling any command you must first create the DLL object. `CLXMSG_CreateObject` returns a handle that you must keep and pass as a parameter to each subsequent call. The handle references a hidden window inside the DLL for capturing Windows messages. Make sure you also destroy the DLL object using `CLXMSG_DestroyObject` before exiting your application. This will prevent a memory leak.

Error handling is standard. A command always returns FALSE if the error parameter is set. The error is then decoded by passing the returned error code `*pnError` to `CLXMSG_BuildErrorText`.

Inside the DLL, each command is based on the Windows messaging technique. This means that commands can potentially timeout if the system is busy. The default timeout of 1 second is sufficient to avoid most timeout situations. If you need to change this value, call `CLXMSG_SetTimeOut` just after creating the DLL object or set the value individually for each command. In the event of a timeout, a function call will block for the timeout period and return FALSE with timeout error code `CLXMSG_ERROR_MSGTIMEOUT`.

Some functions such as `CLXMSG_GetStatus` and `CLXMSG_StartAcquisition` use predefined constants as parameters. e.g. To start an acquisition in view mode use the following code:

```
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_StartAcquisition(m_hClxmsg, CLXMSG_ACQ_START_VIEW, &nError) )
{
    char szError[256] = "";
    CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
    AfxMessageBox(szError, MB_ICONSTOP);
}
```

A complete list of predefined constants can be found in AxClampexMsg.h.

Loading a protocol is very simple. Just pass a pointer to the protocol filename:

```
int nError = CLXMSG_ERROR_NOERROR;
char szPath[] = "C:\\Axon\\Params\\es sine.pro";
if( !CLXMSG_LoadProtocol(m_hClxmsg, szPath, &nError) )
{
    char szError[256] = "";
    CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
    AfxMessageBox(szError, MB_ICONSTOP);
```

```
   }
```

To get started, compile the VC++ example project CLXMSG_TestBed in DevStudio. Run Clampex and then run the test bed. Click "Create DLL", then "Load Protocol" and select a .pro file from the file dialog. Click OK and observe that Clampex changes to the selected protocol. Now click "Record". Clampex will start an acquistion and save data to file.

Example Code

```
#include <afxwin.h>         // MFC core and standard components
#include "AxClampexMsg.h"

//==============================================================================
// FUNCTION: DisplayErrorMsg
// PURPOSE:  Display error as text string
//
void DisplayErrorMsg(HCLXMSG hClxmsg, int nError)
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}

//==============================================================================
// FUNCTION: main
// PURPOSE:  This example shows how to create the DLL handle,
//           load a Clampex protocol, record data, and destroy the handle.
//
int main()
{
   // check the API version matches the expected value
   if( !CLXMSG_CheckAPIVersion(CLXMSG_APIVERSION_STR) )
   {
      AfxMessageBox("Version mismatch: AXCLAMPEXMSG.DLL", MB_ICONSTOP);
      return 0;
   }

   // create DLL handle
   int nError = CLXMSG_ERROR_NOERROR;
   HCLXMSG hClxmsg = CLXMSG_CreateObject(&nError);
   if( !hClxmsg )
   {
      DisplayErrorMsg(hClxmsg, nError);
      return 0;
   }

   // load a protocol
   char szPath[] = "C:\\Axon\\Params\\es sine.pro";
   if( !CLXMSG_LoadProtocol(hClxmsg, szPath, &nError) )
   {
      DisplayErrorMsg(hClxmsg, nError);
      return 0;
   }
   // record data
   if( !CLXMSG_StartAcquisition(hClxmsg, CLXMSG_ACQ_START_RECORD, &nError) )
   {
      DisplayErrorMsg(hClxmsg, nError);
      return 0;
   }

   // destroy DLL handle
   CLXMSG_DestroyObject(hClxmsg);
   hClxmsg = NULL;

   return 0;
}
```

Reference

Commands with similar functions are grouped together.

DLL creation and destruction

BOOL CLXMSG_CheckAPIVersion(LPCSTR pszQueryVersion);

Parameter	**Description**
pszQueryVersion	Pointer to null terminated string containing the version number.

Remarks
Checks the API version matches the expected value: `CLXMSG_APIVERSION_STR`.

Example
```
#include "AxClampexMsg.h"

if( !CLXMSG_CheckAPIVersion(CLXMSG_APIVERSION_STR) )
{
   AfxMessageBox("Version mismatch: AXCLAMPEXMSG.DLL", MB_ICONSTOP);
   return;
}
```

HCLXMSG CLXMSG_CreateObject(int *pnError);

Parameter	**Description**
pnError	Address of error return code.

Remarks
Create the Clampex message handler object and return a 32-bit handle. When successful, the returned handle is non-NULL. You must use this handle to call other CLXMSG functions. Returns NULL if Clampex is not open.

Error Codes
```
CLXMSG_ERROR_OUTOFMEMORY
CLXMSG_ERROR_CLAMPEXNOTOPEN
```

Example
```
#include "AxClampexMsg.h"

int nError = CLXMSG_ERROR_NOERROR;
m_hClxmsg = CLXMSG_CreateObject(&nError);
if( !m_hClxmsg )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(NULL, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

void CLXMSG_DestroyObject(HCLXMSG hClxmsg);

Parameter	**Description**
hClxmsg	Handle to message handler object

Remarks
Destroys the Clampex message handler object. To prevent memory leaks call this before exiting your application.

Example
```
#include "AxClampexMsg.h"

CLXMSG_DestroyObject(m_hClxmsg);
m_hClxmsg = NULL;
```

General

BOOL CLXMSG_SetTimeOut(HCLXMSG hClxmsg, UINT uTimeOutMS, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object
uTimeOutMS	Time out value in milliseconds
pnError	Address of error return code.

Remarks
Sets the time out value for Windows messages between the message handler object and Clampex. Default time out is 1 second.

Example
```
#include "AxClampexMsg.h"

UINT uTimeOut = 2000; // 2 seconds
if( !CLXMSG_SetTimeOut(m_hClxmsg, uTimeOut, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

Acquisition

BOOL CLXMSG_LoadProtocol(HCLXMSG hClxmsg, char *pszFilename, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object
pszFilename	Pointer to a null terminated string containing the protocol filename.
pnError	Address of error return code.

Remarks
Load the Clampex protocol specified by text string `pszFilename`.

Error Codes

`CLXMSG_ERROR_PROTOCOLPATHNOTSET`	The path is incorrectly set.
`CLXMSG_ERROR_PROTOCOLNOTVALID`	The protocol file is not valid and is corrupted.
`CLXMSG_ERROR_PROTOCOLCANNOTLOAD`	The file is not a protocol file and cannot be loaded.
`CLXMSG_ERROR_PROTOCOLCANNOTLOADWHENRECORDING`	A recording is in progress.

Example
```
#include "AxClampexMsg.h"

int nError = CLXMSG_ERROR_NOERROR;
char szPath[] = "C:\\Axon\\Params\\es sine.pro";
if( !CLXMSG_LoadProtocol(m_hClxmsg, szPath, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetStatus(HCLXMSG hClxmsg, UINT *puStatus, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object
puStatus	Address of return Clampex status
pnError	Address of error return code.

Remarks
Determines the current Clampex status. When functions returns TRUE, *puStatus is filled out with one of the following IDs:

`CLXMSG_ACQ_STATUS_IDLE`	Acquisition is not in progress.
`CLXMSG_ACQ_STATUS_DIALOGOPEN`	Dialog is open in Clampex.
`CLXMSG_ACQ_STATUS_TRIGWAIT`	Acquisition is pending waiting for a trigger.
`CLXMSG_ACQ_STATUS_VIEWING`	View-only acquisition is in progress.
`CLXMSG_ACQ_STATUS_RECORDING`	Acquisition is currently recording to disk.
`CLXMSG_ACQ_STATUS_PAUSEVIEW`	Recording has been paused, but display is left running.
`CLXMSG_ACQ_STATUS_PAUSED`	Recording has been paused - display stopped.
`CLXMSG_ACQ_STATUS_DISABLED`	Acquisition has been disabled because of bad parameters.

Example
```
#include "AxClampexMsg.h"
```

```
int nError = CLXMSG_ERROR_NOERROR;
UINT uStatus = 0;
if( !CLXMSG_GetStatus(m_hClxmsg, &uStatus, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetRepeat(HCLXMSG hClxmsg, BOOL bRepeat, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object
bRepeat	Enable Repeat when TRUE, disable when FALSE.
pnError	Address of error return code.

Remarks
Set Clampex Repeat. Clampex will automatically repeat the current acquisition when Repeat is enabled.

Example
```
#include "AxClampexMsg.h"

// enable repeat
int nError   = CLXMSG_ERROR_NOERROR;
BOOL bRepeat = TRUE;
if( !CLXMSG_SetRepeat(m_hClxmsg, bRepeat, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_StartAcquisition(HCLXMSG hClxmsg, UINT uMode, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
uMode	The acquisition mode to start.
pnError	Address of error return code.

Remarks
Start a Clampex acquisition. `uMode` must be filled in with one of the following IDs.

`CLXMSG_ACQ_START_VIEW`	Acquire data with current protocol but do not save to file.
`CLXMSG_ACQ_START_RECORD`	Acquire data with current protocol and save to file.
`CLXMSG_ACQ_START_RERECORD`	Acquire data with current protocol and save over last file.
`CLXMSG_ACQ_START_RERECORD_PROMPT`	Same as above but prompt before overwriting.

Error Codes
```
CLXMSG_ERROR_DIALOGOPEN
CLXMSG_ERROR_UNKNOWNACQMODE
```

Example
```
#include "AxClampexMsg.h"
```

```
// start a Clampex recording
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_StartAcquisition(m_hClxmsg, CLXMSG_ACQ_START_RECORD, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);}
}
```

BOOL CLXMSG_StopAcquisition(HCLXMSG hClxmsg, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object
pnError	Address of error return code.

Remarks
Stop a Clampex acquisition.

Error codes
`CLXMSG_ERROR_STOPIGNOREDWHENIDLE`

Example
```
#include "AxClampexMsg.h"

// stop a Clampex acquisition
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_StartStop(m_hClxmsg, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

Telegraphs

BOOL CLXMSG_GetTelegraphValue(HCLXMSG hClxmsg, UINT uChan, UINT uTelItem, float *pfTelValue, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
uChan	Input channel to query (0-15).
uTelItem	Telegraph item to query.
pfTelValue	Address of return telegraph value.
pnError	Address of error return code.

Remarks
Get the specified telegraph value. `uChan` specifies the analog input channel between 0-15. `uTelItem` specifies the telegraph item and must be one of the following IDs:

`CLXMSG_TEL_CM`	Request the telegraphed whole cell capacitance.
`CLXMSG_TEL_GAIN`	Request the telegraphed gain.
`CLXMSG_TEL_MODE`	Request the telegraphed mode.
`CLXMSG_TEL_FREQUENCY`	Request the telegraphed low pass filter cutoff frequency.
`CLXMSG_TEL_CMDSCALEFACTOR`	Request the telegraphed command scale factor.

Example
```
#include "AxClampexMsg.h"

// get a telegraph value
int nError = CLXMSG_ERROR_NOERROR;
UINT uChan = 0;
float fCm  = 0;
if( !CLXMSG_GetTelegraphValue(m_hClxmsg, uChan, CLXMSG_TEL_CM, &fCm, &nError))
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetTelegraphInstrument (HCLXMSG hClxmsg, UINT uChan, char *pszInstrument, UINT uSize, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
uChan	Input channel to query (0-15).
pszInstrument	Address of return telegraph instrument name.
uSize	Size of return char buffer, must be at least `CLXMSG_TEL_INSTRU_NAMESIZE.`
pnError	Address of error return code.

Remarks
Get the specified telegraph instrument name. `uChan` specifies the analog input channel between 0-15. *`pszInstrument` must be at least `CLXMSG_TEL_INSTRU_NAMESIZE` chars to avoid truncation when filled out with the telegraph instrument name.

Example
```
#include "AxClampexMsg.h"
```

```
// get the telegraph instrument name on the specified input channel
int nError = CLXMSG_ERROR_NOERROR;
char szInstrument[CLXMSG_TEL_INSTRU_NAMESIZE];
if( !CLXMSG_GetTelegraphInstrument(m_hClxmsg, uChan, szInstrument,
sizeof(szInstrument), &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

Membrane Test

BOOL CLXMSG_StartMembTest (HCLXMSG hClxmsg, UINT uOUT, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object
uOut	Membrane test analog output channel ID
pnError	Address of error return code.

Remarks
Open the Clampex membrane test dialog and start the membrane test. `uOut` must be filled as as either:

`CLXMSG_MBT_OUT0`	Membrane Test OUT0
`CLXMSG_MBT_OUT1`	Membrane Test OUT1

Error Codes
```
CLXMSG_ERROR_MEMB_CANNOTSTARTMORETHANONE
CLXMSG_ERROR_MEMB_ALREADYSTARTED
```

Example
```
#include "AxClampexMsg.h"

// start Clampex membrane test on analog output channel 0
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_StartMembTest(m_hClxmsg, CLXMSG_MBT_OUT0, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_StopMembTest (HCLXMSG hClxmsg, UINT uOUT, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object
uOut	Membrane test analog output channel ID
pnError	Address of error return code.

Remarks
Stop the Clampex membrane test and close the membrane test dialog. `uOut` must be filled as as either:

`CLXMSG_MBT_OUT0`	Membrane Test OUT0
`CLXMSG_MBT_OUT1`	Membrane Test OUT1

Error Codes
```
CLXMSG_ERROR_MEMB_ALREADYSTOPPED
```

Example
```
#include "AxClampexMsg.h"

// stop Clampex membrane test on analog output channel 0
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_StopMembTest(m_hClxmsg, CLXMSG_MBT_OUT0, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetMembTestHolding (HCLXMSG hClxmsg, double dHolding, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
dHolding	Membrane test holding level in millivolts.
pnError	Address of error return code.

Remarks
Set the Clampex membrane test holding level in millivolts. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for the holding level to be applied.

Example
```
#include "AxClampexMsg.h"

// set Clampex membrane test holding to 50 mV
int nError = CLXMSG_ERROR_NOERROR;
double dHolding = 50;
if( !CLXMSG_SetMembTestHolding(m_hClxmsg, dHolding, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetMembTestHolding (HCLXMSG hClxmsg, double *pdHolding, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pdHolding	Address of membrane test return holding level in millivolts.
pnError	Address of error return code.

Remarks
Get the Clampex membrane test holding level in millivolts. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for a valid return value.

Example
```
#include "AxClampexMsg.h"

// get Clampex membrane test holding
int nError = CLXMSG_ERROR_NOERROR;
double dHolding = 0;
if( !CLXMSG_GetMembTestHolding(m_hClxmsg, &dHolding, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetMembTestPulseHeight (HCLXMSG hClxmsg, double dPulseHeight, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
dPulseHeight	The membrane test pulse height in millivolts.
pnError	Address of error return code.

Remarks
Set the Clampex membrane test pulse height in millivolts. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for the pulse height to be applied.

Example
```
#include "AxClampexMsg.h"

// set Clampex membrane test pulse height to 10 mV
int nError = CLXMSG_ERROR_NOERROR;
double dPulseHeight = 10;
if( !CLXMSG_SetMembTestPulseHeight(m_hClxmsg, &dPulseHeight, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetMembTestPulseHeight (HCLXMSG hClxmsg, double *pdPulseHeight, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pdPulseHeight	Address of membrane test return pulse height in millivolts.
pnError	Address of error return code.

Remarks
Get the Clampex membrane test pulse height in millivolts. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for a valid return value.

Example
```
#include "AxClampexMsg.h"

// get Clampex membrane test pulse height
int nError = CLXMSG_ERROR_NOERROR;
double dPulseHeight = 0;
if( !CLXMSG_GetMembTestPulseHeight(m_hClxmsg, &dPulseHeight, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_FlushMembTestCache(HCLXMSG hClxmsg, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pnError	Address of error return code.

Remarks
Flush the membrane test cache. This clears the internal buffer for Rt, Ra, Rm, Cm, Tau and Holding values.

Example
```
#include "AxClampexMsg.h"

// flush the internal membrane test cache
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_FlushMembTestCache(m_hClxmsg, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetMembTestCacheSize(HCLXMSG hClxmsg, UINT *puSize, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
puSize	Address of membrane test return cache size
pnError	Address of error return code.

Remarks
Get the membrane test cache size in number of points. One point contains values for Rt, Ra, Rm, Cm, Tau and Holding.

Example
```
#include "AxClampexMsg.h"

// get the internal membrane test cache size
int nError = CLXMSG_ERROR_NOERROR;
UINT uSize = 0;
if( !CLXMSG_GetMembTestCacheSize(m_hClxmsg, &uSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetMembTestCacheMaxSize(HCLXMSG hClxmsg, UINT uMaxSize, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
uMaxSize	The maximum size of the membrane test cache.

pnError	Address of error return code.

Remarks
Set the maximum size of the membrane test cache in number of points. The default is 1000 points. One point contains values for Rt, Ra, Rm, Cm, Tau and Holding.

Example
```
#include "AxClampexMsg.h"

// set the maximum internal membrane test cache size
int nError = CLXMSG_ERROR_NOERROR;
UINT uMaxSize = 0;
if( !CLXMSG_SetMembTestCacheMaxSize(m_hClxmsg, uMaxSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetMembTestCacheData(HCLXMSG hClxmsg, double *pdAvRt, double *pdAvCm, double *pdAvRm, double *pdAvRa, double *pdAvTau, double *pdAvHold, UINT *puCount, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pdAvRt	Address of return average total resistance (Rt).
pdAvCm	Address of return average membrane capacitance (Cm).
pdAvRm	Address of return average membrane resistance (Rm).
pdAvRa	Address of return average access resistance (Ra).
pdAvTau	Address of return average time constant (Tau).
pdAvHold	Address of return average current holding level (Holding).
puCount	Address of return number of points averaged.
pnError	Address of error return code.

Remarks
Get the average value of all data variables in the membrane test cache. You must specify the number of points to average in *`puCount`. To average all points, call `CLXMSG_GetMembTestCacheSize` to determine the number of points currently in the cache. If you set *`puCount` larger than the current cache size, *`puCount` is reset to the current cache size and the average calculated for that size. If the membrane test cannot calculate a valid point, the return value is FALSE and *`pnError` is set. A text description of errors can be obtained from `CLXMSG_BuildErrorText`.

Error codes
```
CLXMSG_ERROR_CACHEISEMPTY
CLXMSG_ERROR_ZEROPOINTSSPECIFIED
CLXMSG_ERROR_MEMB_RESPONSECLIPPED
CLXMSG_ERROR_MEMB_RESPONSERECTIFIED
CLXMSG_ERROR_MEMB_SLOWRISETIME
CLXMSG_ERROR_MEMB_NOPEAKFOUND
CLXMSG_ERROR_MEMB_BADRESPONSE
CLXMSG_ERROR_MEMB_TAUTOOFAST
CLXMSG_ERROR_MEMB_TAUTOOSLOW
CLXMSG_ERROR_MEMB_TOOFEWPOINTS
```

```
CLXMSG_ERROR_MEMB_NOPULSESPECIFIED
CLXMSG_ERROR_MEMB_HOLDINGOUTOFRANGE
CLXMSG_ERROR_MEMB_PULSEOUTOFRANGE
CLXMSG_ERROR_MEMB_INVALIDOUTPUTDAC
```

Example

```
#include "AxClampexMsg.h"

// get the internal membrane test cache data
int    nError     = CLXMSG_ERROR_NOERROR;
UINT   uCacheSize = 0;
double dAvRt      = 0;
double dAvCm      = 0;
double dAvRm      = 0;
double dAvRa      = 0;
double dAvTau     = 0;
double dAvHold    = 0;

// get the current cache size
if( !CLXMSG_GetMembTestCacheSize(m_hClxmsg, &uCacheSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}

// get the average value of this data
if( !CLXMSG_GetMembTestCacheData(m_hClxmsg, &dAvRt, &dAvCm, &dAvRm, &dAvRa,
&dAvTau, &dAvHold, &uCacheSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_ScaleMembTestYAxis(HCLXMSG hClxmsg, UINT uScale, int *pnError);

Parameter	Description
hClxmsg	Handle to message handler object.
uScale	Y axis scale method.
pnError	Address of error return code.

Remarks

Autoscale or fullscale the membrane test Y axis. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for the scale to be set. `uScale` must be filled in as either:

`CLXMSG_MBT_AUTOSCALE`	Autoscale the Membrane Test Y Axis
`CLXMSG_MBT_FULLSCALE`	Fullscale the Membrane Test Y Axis

Example

```
#include "AxClampexMsg.h"

// autoscale the membrane test Y axis
```

```
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_ScaleMembTestYAxis(m_hClxmsg, CLXMSG_MBT_AUTOSCALE, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetMembTestRate(HCLXMSG hClxmsg, double dRate, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
dRate	Set the membrane test rate in Hz.
pnError	Address of error return code.

Remarks
Set the membrane test rate in Hertz. The effective data rate is scaled by the number of points set by `CLXMSG_SetMembTestAveraging`. For instance, if the rate is 500 Hz and the number of membrane test averaging points is 10, the data rate is 50 Hz. The maximum rate will be determined by the digiziter selected in Clampex. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for the rate to be applied.

Example
```
#include "AxClampexMsg.h"

// set the membrane test rate to 500 Hz
int nError = CLXMSG_ERROR_NOERROR;
double dRate = 500;
if( !CLXMSG_SetMembTestRate (m_hClxmsg, dRate, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetMembTestRate(HCLXMSG hClxmsg, double *pdRate, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pdRate	Address of membrane test return rate in Hz.
pnError	Address of error return code.

Remarks
Get the membrane test rate in Hertz. The effective data rate is scaled by the number of points set by `CLXMSG_SetMembTestAveraging`. For instance, if the rate is 500 Hz and the number of membrane test averaging points is 10, the data rate is 50 Hz. The maximum rate will be determined by the digiziter selected in Clampex. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for a valid return value.

Example
```
#include "AxClampexMsg.h"

// get the membrane test rate
int nError = CLXMSG_ERROR_NOERROR;
double dRate = 0;
if( !CLXMSG_GetMembTestRate(m_hClxmsg, &dRate, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetMembTestAveraging (HCLXMSG hClxmsg, BOOL bAveraging, UINT uNumEdges, int *pnError);

Parameter	Description
hClxmsg	Handle to message handler object.
bAveraging	Set the membrane test enable.
uNumEdges	Set the number of pulse edges to average;
pnError	Address of error return code.

Remarks
Set membrane test averaging parameters. Set bAveraging to control the "Averaging" check box and uNumEdges to control the "Edges per average" edit box in the membrane test dialog. If bAveraging is TRUE and uNumEdges is 10, each point received by the membrane test cache represents the average of 10 membrane test calculations. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for changes to be applied.

Example
```
#include "AxClampexMsg.h"

// enable membrane test averaging for 10 pulse edges
int nError = CLXMSG_ERROR_NOERROR;
BOOL bAveraging = TRUE
UINT uNumEdges  = 10;
if( !CLXMSG_SetMembTestAveraging(m_hClxmsg, bAveraging, uNumEdges, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetMembTestAveraging (HCLXMSG hClxmsg, BOOL *pbAveraging, UINT *puNumEdges, int *pnError);

Parameter	Description
hClxmsg	Handle to message handler object.
pbAveraging	Address of membrane test return averaging enable state.
puNumEdges	Address of membrane test return number of pulse edges per average.
pnError	Address of error return code.

Remarks
Get membrane test averaging parameters. *pbAveraging is the current state of the "Averaging" check box and *puNumEdges is the current value of the "Edges per average" edit box in the membrane test dialog. The membrane test dialog must be opened either manually or with `CLXMSG_StartMembTest` for valid return values.

Example
```
#include "AxClampexMsg.h"

// enable membrane test averaging for 10 pulse edges
int nError = CLXMSG_ERROR_NOERROR;
BOOL bAveraging = FALSE;
UINT uNumEdges  = 0;
if( !CLXMSG_GetMembTestAveraging(m_hClxmsg, &bAveraging, &uNumEdges, &nError)
)
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

Seal Test

BOOL CLXMSG_SetSealTestHolding (HCLXMSG hClxmsg, double dHolding, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
dHolding	The seal test holding level in millivolts.
pnError	Address of error return code.

Remarks
Set the Clampex seal test holding level in millivolts. The seal test dialog must be opened manually for the holding level to be applied.

Example
```
#include "AxClampexMsg.h"

// set Clampex seal test holding to 50 mV
int nError = CLXMSG_ERROR_NOERROR;
double dHolding = 50;
if( !CLXMSG_SetSealTestHolding(m_hClxmsg, dHolding, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetSealTestHolding (HCLXMSG hClxmsg, double *pdHolding, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pdHolding	Address of seal test return holding level in millivolts.
pnError	Address of error return code.

Remarks
Get the Clampex seal test holding level in millivolts. The seal test dialog must be opened manually for a valid return value.

Example
```
#include "AxClampexMsg.h"

// get Clampex seal test holding
int nError = CLXMSG_ERROR_NOERROR;
double dHolding = 0;
if( !CLXMSG_GetSealTestHolding(m_hClxmsg, &dHolding, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetSealTestPulseHeight (HCLXMSG hClxmsg, double dPulseHeight, int *pnError);

Parameter	Description
hClxmsg	Handle to message handler object.
dPulseHeight	The seal test pulse height in millivolts.
pnError	Address of error return code.

Remarks
Set the Clampex seal test pulse height in millivolts. The seal test dialog must be opened manually for the pulse height to be applied.

Example
```
#include "AxClampexMsg.h"

// set Clampex seal test pulse height to 10 mV
int nError = CLXMSG_ERROR_NOERROR;
double dPulseHeight = 10;
if( !CLXMSG_SetSealTestPulseHeight(m_hClxmsg, &dPulseHeight, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetSealTestPulseHeight (HCLXMSG hClxmsg, double *pdPulseHeight, int *pnError);

Parameter	Description
hClxmsg	Handle to message handler object.
pdPulseHeight	Address of seal test return pulse height in millivolts.
pnError	Address of error return code.

Remarks
Get the Clampex seal test pulse height in millivolts. The seal test dialog must be opened manually for a valid return value.

Example
```
#include "AxClampexMsg.h"
```

```
// get Clampex seal test pulse height
int nError = CLXMSG_ERROR_NOERROR;
double dPulseHeight = 0;
if( !CLXMSG_GetSealTestPulseHeight(m_hClxmsg, &dPulseHeight, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_FlushSealTestCache(HCLXMSG hClxmsg, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pnError	Address of error return code.

Remarks
Flush the seal test cache. This clears the internal seal resistance (Rs) buffer.

Example
```
#include "AxClampexMsg.h"

// flush the internal seal test cache
int nError = CLXMSG_ERROR_NOERROR;
if( !CLXMSG_FlushSealTestCache(m_hClxmsg, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetSealTestCacheSize(HCLXMSG hClxmsg, UINT *puSize, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
puSize	Address of seal test return cache size
pnError	Address of error return code.

Remarks
Get the seal test cache size in number of points. One point contains one seal resistance (Rs) value.

Example
```
#include "AxClampexMsg.h"

// get the internal seal test cache size
int nError = CLXMSG_ERROR_NOERROR;
UINT uSize  = 0;
if( !CLXMSG_GetSealTestCacheSize(m_hClxmsg, &uSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_SetSealTestCacheMaxSize(HCLXMSG hClxmsg, UINT uMaxSize, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
uMaxSize	The maximum size of the seal test cache.
pnError	Address of error return code.

Remarks
Set the maximum size of the seal test cache in number of points. The default is 1000 points. One point contains one seal resistance (Rs) value.

Example
```
#include "AxClampexMsg.h"

// set the maximum internal seal test cache size
int  nError   = CLXMSG_ERROR_NOERROR;
UINT uMaxSize = 0;
if( !CLXMSG_SetSealTestCacheMaxSize(m_hClxmsg, uMaxSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

BOOL CLXMSG_GetSealTestCacheData(HCLXMSG hClxmsg, double *pdAvRs, UINT *puCount, int *pnError);

Parameter	**Description**
hClxmsg	Handle to message handler object.
pdAvRs	Address of return average seal resistance (Rs).
puCount	Address of return number of points averaged.
pnError	Address of error return code.

Remarks
Get the average value of seal resistance in the seal test cache. You must specify the number of points to average in *`puCount`. To average all points, call `CLXMSG_GetSealTestCacheSize` to determine the number of points currently in the cache. If you set *`puCount` larger than the current cache size, *`puCount` is reset to the current cache size and the average calculated for that size.

Example
```
#include "AxClampexMsg.h"

// get the current cache size
int  nError     = CLXMSG_ERROR_NOERROR;
UINT uCacheSize = 0;
if( !CLXMSG_GetSealTestCacheSize(m_hClxmsg, &uCacheSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}

// get the average seal resistance
double dAvRs = 0;
if( !CLXMSG_GetSealTestCacheData(m_hClxmsg, &dAvRs, &uCacheSize, &nError) )
{
   char szError[256] = "";
   CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
   AfxMessageBox(szError, MB_ICONSTOP);
}
```

Error Handling

BOOL CLXMSG_BuildErrorText (HCLXMSG hClxmsg, int nErrorNum, LPSTR sTxtBuf, UINT uMaxLen);

Parameter	**Description**
hClxmsg	Handle to message handler object.
nErrorNum	The error code.
sTxtBuf	Char buffer to return error string
uMaxLen	Size of char buffer

Remarks
Return the error as a text string.

Example
```
#include "AxClampexMsg.h"

// get error as a string
char szError[256] = "";
int nError = CLXMSG_ERROR_PROTOCOLPATHNOTSET;
CLXMSG_BuildErrorText(m_hClxmsg, nError, szError, sizeof(szError));
AfxMessageBox(szError, MB_ICONSTOP);
```