

Using of ANum.dll

Presentation

This dynamic library allows to remote FrameDecoder.exe, GeneMod.exe and GeneRF.exe applications from another program written in another language with dll capabilities.

The DLL manages the Shared Memory, opens the applications and manages the information exchange.

The principle is simple : by the intermediary of DLL functions, you can send a command to an application (until 8 different applications can be opened at the same time).

The concerned application receives the command, treats it and sends back a response. The concerned application can also manages an event to the master program at the end of an action for example.

Terms

function : DLL function

command : command sent (text form) to an application with ANumMasterWrite function

response : application answer (text form)

Functions list

ANumMasterOpen,
ANumMasterWrite,
ANumMasterRead,
ANumMasterClose,
ANumMasterEvent.

• ANumMasterOpen

Function :

Opens a application and creates the associated shared memory. The application runs invisible by default. To show it, send "Show" command with ANumMasterWrite.

Note : before send the first command, wait for the necessary time for the initialization of the application.

Syntax :

```
char ANumMasterOpen(char index, char *program)
```

Parameters :

index : application number (0..7). Keep this index during all application execution.

program : application name to open with the full path

"C:\\ANum\\FrameDecoder.exe" : for FrameDecoder.exe

" C:\\ANum\\GeneMod.exe" : for GeneMod.exe

Return :

00=OK

01=error on CreateFileMapping

02=error on MapViewOfFile

03=error on CreateProcess

04=index busy (ANumMasterOpen already made)

Example :

```
ret=ANumMasterOpen(0, ".\\slave.exe");
```

Note :

The shared memory is created automatically from the name of program by adding "SM".

• ANumMasterWrite

Function :

Sends a command to the application. This function waits the application responds during **2 seconds** maximum. Beyond this time, the error 03 is returned.

To read the response, use ANumMasterRead function.

Syntax :

char ANumMasterWrite (char index, char *command)

Parameters :

index : application number (0..7) used for ANumMasterOpen

command : command (text form) finishing with a null character . Respect the case, and don't use space character. (see just below the command list according to the application).

Return :

00=OK (there is a response from application, to read with ANumMasterRead)

01=no response from the application

02=no application (closed)

03=Index not attributed (you must do a ANumMasterOpen before)

Example :

```
ret=ANumMasterWrite(0, "Ask");
```

Available commands with ANumMasterWrite :

FrameDecoder Commands :

- "Hide" : hides the application and returns "OK" if no error
- "Show" : hides the application and returns "OK" if no error
- "Close" : closes the application (use instead ANumMasterClose)
- "Ask" : gets a line from Hexaview window. If there is no line, returns "(nothing)"
- "LoadConfiguration>file.cfg" : loads a configuration and return "OK" if the file exists, else returns "(File not found)"
- "Run" : starts decoding
- "Com>port" : changes COM port number. In no error, returns "OK" else returns "(Error port)"
- "Stop" : stops decoding

GeneMod Commands :

- "Hide" : hides the application and returns "OK" if no error
- "Show" : hides the application and returns "OK" if no error
- "Close" : closes the application (use instead ANumMasterClose)
- "LoadConfiguration>file.cfg" : loads a configuration and return "OK" if the file exists, else returns "(File not found)"
- "SendF>frame_name" : sends a defined frame and returns "OK" if the frame exists.
- "Execute>script_name" : executes a script defined in the script and returns "OK" if the file exists.
- "SendP>Content,Repetition,Duration,Frequency,Ratio" : send a frame with its parameters. The value of each parameter (except Content) must be the same of one in the GeneMod ListBoxes. Returns "OK" or "(Syntax error)" in case of error on one of the parameters.
- If a frame is beeing sent, returns "(Not Ready)". Then, you must send a "Stop" before send another "SendP" command.
- "Com>port" : changes the COM port number. Possible values are 1..12 .Returns "OK" or "(Error port)"
- "Stop" : stops the frame sending or the script execution.

Examples :

"SendF>GM-Force_TX"

"SendP>L32(AAAAAAAAh) N18(111000101100110010b) M16(615Eh) M32(13C66C39h) R5(0b),130,128,125.000,50"

"Com>4"

GeneRF commands :

- "Hide" : hides the application and returns "OK" if no error
- "Show" : hides the application and returns "OK" if no error
- "Close" : closes the application (use instead ANumMasterClose)
- "LoadConfiguration>file.cfg" : loads a configuration and return "OK" if the file exists, else returns "(File not found)"
- "SendF>frame_name" : sends a defined frame and returns "OK" if the frame exists.
- "Execute>script_name" : execute a script defined in the script and returns "OK" if the file exists.
- "SendP>Content,Duration,SettleTime,Modulation" : send a frame with its parameters. The value of each parameter (except Content) must be the same of one in the ListBox. Returns "OK" or "(Syntax error)" in case of error on one of the parameters.
- If a frame is sending, returns "(Not Ready)". Then, you must send a "Stop".
- "Com>port" : changes the COM port number. Possible values are 1..12 .Returns "OK" or "(Error port)"
- "Stop" : stops the frame sending or script execution.

Examples :

"SendF>GM-Force_TX"

"SendP>R100(1001b) I32(0001h) I32(1234h) I32(5678h) I8(00h),52.2,3.6,ASK"

• **ANumMasterRead**

Function :

Reads the last response of the application.

Syntax :

char* ANumMasterRead (char index)

Parameter :

index : application number (0..7) used for ANumMasterOpen

Return :

A pointer on the response finishing with a null character.

This function must be called if the ANumMasterWrite function has return value=00

Example :

Lines->Add(ANumRead(0));

• ANumMasterClose**Function :**

Sends "Close" command to the application and frees the shared memory.

The launched application is closed.

Once closed, the application can be relaunched with the ANumMasterOpen function.

Syntax :

char* ANumMasterClose (char index)

Parameter :

index : application number (0..7) used for ANumMasterOpen

Return :

00=OK (there is a response from application, to read with ANumMasterRead)

01=no response from the application

02=no application (closed)

03=Index not attributed (you must do a ANumMasterOpen before)

Example :

```
ret=ANumMasterClose(0);
```

• **ANumMasterEvent**

Function :

Programs the DLL so that it calls a specified function in your program when it receives an event.

Syntax :

```
char ANumMasterEvent(char index, void *function)
```

Parameter :

index : application number (0..7) used for ANumMasterOpen

function : called function with this prototype : void FDCallback()

Return :

00=OK (there is a response from application, to read with ANumMasterRead)

03=Index not attributed (you must do a ANumMasterOpen before)

Example :

```
ret=ANumMasterEvent(0, &FDCallback);
```

Available events :

From **FrameDecoder** :

- when a new line is displayed in HexaView. You must call ANumMasterRead function to read the line with text format.

From **GeneMod** :

- End of frame sending. ANumMasterRead will return "Sent".

- End of script execution. ANumMasterRead will return "Executed".

From **GeneRF** :

- End of frame sending. ANumMasterRead will return "Sent".

- End of script execution. ANumMasterRead will return "Executed".

DLL Using

The functions are type `__stdcall`, so there are compatible with the WIN32 standard passage of parameters (restoration of the stack by the called function, parameters pushed by the right to the left in the stack, transmission of the number and the appropriate types of arguments).

This example is created with Borland C++ Builder.

It shows how load the DLL and use it to execute FrameDecoder.

```
//-----
// DECLARATION OF DLL FUNCTIONS
//-----
typedef char __stdcall (*TANumOpen) (char index, char *program);
typedef char __stdcall (*TANumWrite) (char index, char *command);
typedef char* __stdcall (*TANumRead) (char index);
typedef char __stdcall (*TANumClose) (char index);
typedef char __stdcall (*TANumEvent) (char index, void* function);
TANumOpen ANumOpen;
TANumWrite ANumWrite;
TANumRead ANumRead;
TANumClose ANumClose;
TANumEvent ANumEvent;
HINSTANCE hDLL;

//-----
// INITIALIZATION OF DLL FUNCTIONS
//-----
__fastcall TfrmMasterMain::TfrmMasterMain(TComponent* Owner)
: TForm(Owner)
{
    // Dynamic loading
    hDLL=LoadLibrary("ANum.dll");
    if (hDLL)
    {
        // Function pointers
        ANumOpen=(TANumOpen)GetProcAddress(hDLL, "ANumMasterOpen");
        if (ANumOpen==NULL) ShowMessage("ANumOpen not found");
        ANumWrite=(TANumWrite)GetProcAddress(hDLL, "ANumMasterWrite");
        if (ANumWrite==NULL) ShowMessage("ANumWrite not found");
        ANumRead=(TANumRead)GetProcAddress(hDLL, "ANumMasterRead");
        if (ANumRead==NULL) ShowMessage("ANumRead not found");
        ANumClose=(TANumClose)GetProcAddress(hDLL, "ANumMasterClose");
        if (ANumClose==NULL) ShowMessage("ANumClose not found");
        ANumEvent=(TANumEvent)GetProcAddress(hDLL, "ANumMasterEvent");
        if (ANumEvent==NULL) ShowMessage("ANumEvent not found");

        // Run FrameDecoder
        ANumOpen(0, "C:\\ANum\\FrameDecoder.exe");
        Sleep(100); // Waits 100ms before send the first command to application
        ANumWrite(0,"Show"); }

        // ... Using
        Sleep(1000);

        // Close DLL
        ANumClose(0);

        // Frees library
        FreeLibraryAndExitThread(hDLL, 00);
    }

    // In case of error
    else
    {
        ShowMessage("ANum.dll not found");
        Close();
    }
}
```