Using the Logger32 External Interface

Bob Furzer K4CY

A series of Windows messages have been coded that allow message exchange between Logger32 and a user-developed application, and data transfer using ADIF formatted strings.

The first step in establishing communication between your application and Logger32 your application must identify the hWnd of the Logger32 MDI:

```
Dim L32hWnd as long

L32hWnd = FindWindow(vbNullString, "Logger32")
```

If Logger32 is not running, your application must retry periodically until Logger32 is running.

Second, your application must register/identify an unused Windows message (Logger32 supports up to 5 simultaneous external applications):

```
Dim L32Msg as long

L32Msg = RegisterWindowMessage("Logger32 1").

PostMessage L32hWnd, L32Msg, 1, GhW ‘ GhW is the hWnd of your main form
```

Logger32 will reply to you with a message number 0. If the L32Msg you send is in use by another external application, the lParam will be set to 0. If the L32Msg is unused, then the lParam will be set to 1. If your connect request was rejected (lParam is 0), then your application must retry using a different L32Msg:

```
L32Msg = RegisterWindowMessage("Logger32 2").

PostMessage L32hWnd, L32Msg, 1, GhW ‘ GhW is the hWnd of your main form
```

Logger32 currently supports up to 5 external interface applications. Your application can retry the connection using a RegisteredWindowMessage “Logger32 1” to “Logger32 5”.

When your application has successfully connected to Logger32 (you receive a message number 0 with an lParam of 1), you must now send a message number 2 to Logger32 with an lParam of the TextBox.hWnd Logger32 is to send text strings to. If your application does not want unsolicited data from Logger32, then you set the lParam to 0:

```
PostMessage L32hWnd, L32Msg, 2, TextBox.hWnd
```

Immediately following the message 2 to Logger32 you have the option to send a message 3 to Logger32. Setting the lParam to 0 (or simply not sending a message 3) will tell Logger32 to not send any DX Spot information to your application (I suspect most applications will not want to receive DX Spots from Logger32). Setting the lParam to 1:

```
PostMessage L32hWnd, L32Msg, 3, 1
```

will tell Logger32 to send ADIF formatted text to your application for each DX Spot received by Logger32. If you have previously set the message 2 lParam to 0 (no TextBox hWnd provided to Logger32) and you set the message 3 lParam to 1 (telling Logger32 to send you DX Spot information) Logger32 will complain, generate a warning message and turn off the request.

Logger32 will respond by doing three things:

a) Logger32 sends your application a message number 3. The lParam of this message is the hWnd of a TextBox in Logger32 your application sends ADIF formatted text strings to.

b) Logger32 sends your application a message 100. The lParam of this message is the radio frequency (in Hz).

c) If your application has OK'd the receipt of unsolicited text strings from Logger32 (you sent a TextBox hWnd as the lParam of your message number 2), Logger32 will send your application a WM_SETTEXT message and put the current Radio Mode in the TextBox you have identified. The format of the text is (say) <APP_RADIO_MODE:3>SSB

The basic connection and synchronization between Logger32 and your application is complete. Logger32 will respond to the following additional messages from your application:

4. lParam is 0. Your application tells Logger32 it has stopped. Logger32 will free up the RegisteredWindowMessage your application was using.

5. lParam is 0. Your application tells Logger32 to shut down.

6. lParam is 0. Your application relinquishes PTT control. Logger32 assumes PTT control and initialized PTT ports/keying lines.

7. lParam is 0. Your application assumes control of the PTT.

8. lParam is 0. Your application tells Logger32 to key the PTT.

9. lParam is 0. Your application tells Logger32 to unkey the PTT.

10. lParam is the hWnd of a TextBox in your application to write to. Logger32 will respond with an ADIF formatted text string in the following format <APP_SET_FREQ_MODE:27>18132.012|CW The frequency is in KHz and the decimal separator will be in the correct format for the PCs regional settings.

11. lParam is 0. Your application tells Logger32 to release CAT control of the Radio.

12. lParam is 0. Your application tells Logger32 to take CAT control of the Radio.

13. lParam is 0. If the Logger32 Mode controlled by Sound Card option is checked, then <APP_FORCE_MODE:x> messages from your application will change Logger32 Mode.

14. lParam is 0. Disables the feature turned on by message 13. Remember to turn this off when your application closes.

15. If lParam is 1 then a flag is set so that Logged QSOs will be marked for QSLing. If lParam is 0 then the flag is turned off.

16. If lParam is 1 then a flag is set so that Logged QSOs will be marked for eQSLing. If lParam is 0 then the flag is turned off.

17. If lParam is 1 then a flag is set so that Logged QSOs will be marked for LoTW. If lParam is 0 then the flag is turned off.

Logger32 will send the following additional messages to your application:

100. The lParam contains the radio frequency in Hz. This message is sent whenever the radio frequency changes.

No more are currently coded … all you have to do is ask.

Data is exchanged between your application and Logger32 by writing ADIF formatted text strings to identified TextBoxes. It is suggested/recommended that the user utilize the TextBox_Change event as a trigger to process the ADIF data received.

Logger32 recognizes the following ADIF strings:

<EOR> If received without additional text, Logger32 will clear all current entries in the Logbook Entry Window.

<APP_TAB> If received without additional text, this simulates moving the focus from the Callsign field of the Logger32 Logbook Entry Window. Automatic functions (like QRZ lookup, Previous QSOs lookup, auto-populate of data from previous QSOs, etc. are triggered.

<CALL:x> Callsign is placed on the Logbook Entry window.

<RST_RCVD:x> The RST Received is placed on the Logbook Entry window.

<RST_SENT:x> The RST Sent is placed on the Logbook Entry window.

<NAME:x> The name is placed on the Logbook Entry window.

<APP_TIME_ON:x> is in the Microsoft timestamp format (example: 39470.6737384259) and sets the ADIF QSO_DATE and TIME_ON fields.

<APP_TIME_OFF:x> is in the Microsoft timestamp format (example: 39470.6737384259) and sets the ADIF TIME_OFF field.

<APP_QSL:1>Y sets the Logger32 QSL flag on. Any other character(s) turns the flag off.

<APP_eQSL:1>Y sets the Logger32 eQSL flag on. Any other character(s) turns the flag off.

<APP_LoTW:1>Y sets the Logger32 LoTW flag on. Any other character(s) turns the flag off.

<COMMENT:x> the comment is placed on the Logbook Entry window.

<QTH:x> the QTH is placed on the Logbook Entry window.

<ADDRESS:x> the address is placed on the Logbook Entry window.

<STATE:x> if the user has assigned STATE as a user field on the Logbook Entry window then the Primary Administrative Subdivision is placed on the Logbook Entry window.

<CNTY:x> if the user has assigned CNTY as a user field on the Logbook Entry window then the Secondary Administrative Subdivision is placed on the Logbook Entry window.

<GRIDSQUARE:x> if the user has assigned GRIDSQUARE as a user field on the Logbook Entry window then the gridsquare is placed on the Logbook Entry window.

<IOTA:x> if the user has assigned IOTA as a user field on the Logbook Entry window then the IOTA is placed on the Logbook Entry window.

<STX:x> if the user has assigned STX as a user field on the Logbook Entry window then the transmitted serial number is placed on the Logbook Entry window.

<SRX:x> if the user has assigned SRX as a user field on the Logbook Entry window then the received serial number is placed on the Logbook Entry window.

<QSL_VIA:x> if the user has assigned QSL_VIA as a user field on the Logbook Entry window then the QSL information is placed on the Logbook Entry window.

<SAT_NAME:x> if the user has assigned SAT_NAME as a user field on the Logbook Entry window then the satellite name is placed on the Logbook Entry window.

<SAT_MODE:x> if the user has assigned SAT_MODE as a user field on the Logbook Entry window then the satellite mode is placed on the Logbook Entry window.

<PROP_MODE:x> if the user has assigned PROP_MODE as a user field on the Logbook Entry window then the propagation mode is placed on the Logbook Entry window.

<FREQ_RX:x> if the user has assigned FREQ_RX as a user field on the Logbook Entry window then the receive frequency is placed on the Logbook Entry window.

<TEN_TEN:x> if the user has assigned TEN_TEN as a user field on the Logbook Entry window then the 10x10 number is placed on the Logbook Entry window.

<USER_1:x> if the user has assigned USER_1 as a user field on the Logbook Entry window then the user defined data is placed on the Logbook Entry window.

<USER_2:x> if the user has assigned USER_2 as a user field on the Logbook Entry window then the user defined data is placed on the Logbook Entry window.

<USER_3:x> if the user has assigned USER_3 as a user field on the Logbook Entry window then the user defined data is placed on the Logbook Entry window.

<APP_LOGQSO:x> will log data currently on the Logbook Entry window. This may be sent as standalone data, or part of a complete QSO to be logged..

<FREQ:x> if the Logger32 Radio Type is set to none, then this data (in KHz) will simulate a change of radio frequency in Logger32.

<MODE:x> if the data received does not match the mode of the Logbook Entry window, a warning message is generated.

<APP_FORCE_MODE:x> This will change the mode in the Logbook Entry window. The message is intended to be used when Logger32 does not have control of the radio. If Logger32 has control of the radio, the Mode you have forced may be overwritten by the mode from the next poll.

<APP_SET_FREQ_MODE:x> Logger32 will set the radio frequency and radio mode to the data

received. The format is (say) <APP_SET_FREQ_MODE:14>14003.451|CW-R Note that the frequency is in KHz.

<APP_CLICK_DXSPOT:x> This simulates clicking on a DX Spot in Logger32. Your application must pass both the frequency (in KHz) and the DX station callsign. The format is (say) <APP_CLICK_DXSPOT:13>14003.01|K4CY.

Logger32 sends the following ADIF strings to your application (if you have allowed unsolicited data to be sent):

<CALL:x> is sent if the Callsign is changed on the Logbook Entry window

<RST_SENT:x> is sent when the RST_SENT field is changed on the Logbook Entry window

<RST_RCVD:x> is sent when the RST_RCVD field is changed on the Logbook Entry window

<NAME:x> is sent when the NAME field is changed on the Logbook Entry window

<MODE:x> is sent when the MODE of the Logbook Entry window is changed.

<APP_RADIO_MODE:x> is sent when the radio mode is changed.

<APP_DXSPOT_CALLSIGN:x> is sent when a DX Spot is received.

<APP_DXSPOT_FREQ:x> is sent when a DX Spot is received.

<APP_DXSPOT_BAND:x> is sent when a DX Spot is received. The band is derived from the Logger32 BandPlan.

<APP_DXSPOT_MODE:x> is sent when a DX Spot is received. The operation mode (i.e. SSB, CW, RTTY, etc.) mode is derived from the Logger32 BandPlan.

<APP_DXSPOT_COLOR:x> is sent when a DX Spot is received if the DX Spot is highlighted.

<EOR> is sent when Logger32 clears the Logbook Entry window.

If you write an application or interface to Logger32, please check the code handles frequency strings where the decimal separator may be either a period or a comma (depending on the users PC Regional Settings). Also, please ensure you handles the connect sequence correctly so that you don't tie up more than one RegisteredWindowMessage. The sample code provides a working sample.

Finally, please make sure you release any resources (including your assigned RegisteredWindowMessage when your application closes.