Application Note AN071

Service-Requests und Errorhandler unter Visual Basic

Autor: JD Datei: AN071.doc (2 Seiten)

Problematik

Bei der Verwendung von Service-Requests sowie des Errorhandlers unter Visual Basic 5.0 oder 6.0 unter den 32-Bit Windows-Betriebssystemen sind einige Besonderheiten gegenüber anderen Programmiersprachen zu beachten.

Der Empfang eines Service-Requests von einer SORCUS-Karte oder ein von der Bibliothek erkannter Fehler führen im PC-Programm normalerweise zum Aufruf von Anwender-Routinen, die vorher mit *mlx_set_service* bzw. *mlx_set_error_handling* installiert worden sein müssen.

Die Bibliothek MLXW32 startet intern einen zweiten Thread, der die Aufgabe hat, Service- und Error-Requests vom Treiber entgegen zu nehmen. Bei einem Request ruft dieser Thread die Anwender-Service-Prozedur bzw. den Errorhandler auf. Visual Basic erlaubt zwar seit der Version 5.0 den Aufruf von Callback-Funktionen, diese dürfen jedoch nicht aus einem anderen Thread heraus erfolgen, da Visual Basic nicht Multi-Threading fähig ist. Der bisherige Mechanismus der Request-Behandlung führt daher zu Schutzverletzungen.

Lösung des Problems für die SRQ-Service-Routine

Um das Problem zu umgehen, wurde speziell für Visual Basic ein anderer Mechanismus in der MLXW32 implementiert. Anstatt die Anwender-Routine direkt aufzurufen sendet der SRQ-Thread der Bibliothek nun eine definierte Windows-Nachricht an ein Fenster der Visual Basic Anwendung. Das empfangene SRQ-Wort wird im SRQ-Thread ausgelesen und zusammen mit der Kartennummer der Karte, die den SRQ gesendet hat, an die Windows-Nachricht angehängt.

Die Visual Basic Anwendung hat die Aufgabe, sich in die Nachrichten-Behandlung des Fensters einzuhängen.

Folgende Programmierschritte sind erforderlich:

- Aufruf der Funktion *mlx_set_service*. Anders als im Handbuch beschrieben wird ihr aber nicht die Adresse der Service-Prozedur übergeben, sondern der Wert, den Windows für die zu sendende Nachricht verwenden soll (*WM_SRQ_RECEIVED = WM_USER + x). WM_USER* hat den Wert 400h, der verwendete Wert sollte größer sein und nicht von anderen Absendern benutzt werden. Die Bibliothek speichert sich intern das Handle des zu diesem Zeitpunkt aktiven Fensters. Es dient ihr später als Adressat für die Windows-Nachricht.

- Setzen der Behandlungs-Routine für das Fenster (sogenanntes Sub-Classing). Zunächst sollte in einer globalen LONG-Variablen (*WndHandle*) das Handle des aktiven Fensters gespeichert werden:
  ```
  WndHandle = GetActiveWindow
  ```
  Anschließend wird mit der Windows API-Funktion *SetWindowLong* die Fenster-Prozedur auf die eigene Prozedur *MyServiceProc* umgesetzt:
  ```
  OldWndProc = SetWindowLong(WndHandle,GWL_WNDPROC,AddressOf MyServiceProc)
  ```

Wichtig ist, die ursprüngliche Fenster-Prozedur in einer globalen LONG-Variablen (*OldWndProc*) zu retten, um sie beim Beenden des Programmes wieder zurücksetzen zu können.

- Die eigene Fenster-Prozedur muß folgenden Aufbau haben:

```
Function MyServiceProc(ByVal hwnd As Long, ByVal uMsg As Long,
                       ByVal wParam As Long, ByVal lParam As Long) As Long
  If uMsg = WM_SRQ_RECEIVED Then
    ...                ´Anwendungs-spezifische SRQ-Auswertung
  Else
    MyServiceProc = CallWindowProc(OldWndProc, hwnd, uMsg, wParam, lParam)
  End If
End Function
```

In den Parametern werden folgende Informationen übergeben:

hWnd: Handle des Fensters
uMsg: Wert der Windows-Nachricht
wParam: Nummer der Karte, die den SRQ abgesendet hat
lParam: das von der SORCUS-Karte empfangene SRQ-Wort

Andere Nachrichten als die vordefinierte müssen mit Hilfe der Windows API Funktion *CallWindowProc* an die ursprüngliche Fenster-Prozedur weitergeleitet werden.

In der Behandlung des empfangenen SRQs darf die sonst in der SRQ-Service-Routine vorgeschriebene Funktion *mlx_get_message* **nicht** aufgerufen werden – der Wert des SRQ-Wortes wurde bereits als Parameter übergeben.

- Wichtig ist, beim Beenden des Programmes (am besten im *Form_Unload* Event) die ursprüngliche Fenster-Prozedur wieder herzustellen:
  `dummy = SetWindowLong(WndHandle, GWL_WNDPROC, oldaddress)`

Fehlerbehandlung

Die Signalisierung von Fehlern unter Visual Basic erfolgt auf die selbe soeben beschriebene Weise. Die Initialisierung erfolgt mit der Funktion *mlx_set_error_handling*, der der Wert der für Fehlermeldungen zu verwendenden Windows-Nachricht (*WM_ERROR_HANDLER*) zu übergeben ist. Fehlermeldungen können in der selben Fenster-Prozedur wie SRQs behandelt werden, die Unterscheidung erfolgt anhand des Wertes der Windows-Nachricht. Im Parameter *lParam* wird der Fehlerwert (Klasse und Code) übergeben.

Das Setzen der Fehlerbehandlungs-Prozedur darf nur mit *mlx_set_error_handling* erfolgen, nicht in *mlx_reset* oder *mlx_start*!

Die Deklarationen der benötigten API-Funktionen befinden sich in der mitgelieferten Importbibliothek *MLXBIB.BAS* ebenso wie die Konstanten *GWL_WNDPROC*, *WM_SRQ_RECEIVED* und *WM_ERROR_MSG*. Die letzteren beiden können bei Bedarf verändert werden.

Die MLXW32-Bibliothek enthält ab der Version 7C.003 die beschriebene Funktionalität.