# LabVIEW Funktionsbibliothek

**Installation und Einführung**

systec Elektronik und Software GmbH

**Systec Elektronik und Software GmbH**
Nottulner Landweg 90
D-48161 Münster-Roxel

Telefon +49-2534-8001-70
Telefax +49-2534-8001-77

LabVIEW Funktionsbibliothek für Xemo-Steuerungen
Doku-Nr. 767-22-3.4
Stand: 05 2017

Originalanleitung

# Inhalt

# 1 Wichtige Symbole in dieser Anleitung

**Hinweis**

Lesen Sie Passagen, welche mit diesem Symbol gekennzeichnet sind, bitte auf jeden Fall. Sie erhalten hier wichtige Informationen zum Umgang mit dieser Anleitung, und Voraussetzungen oder Grenzen für die Nutzung der LabVIEW VIs.

**Tipp**

Erfahren Sie in so gekennzeichneten Abschnitten zusätzlich Wissenswertes und praktische Tipps.

**[SYSTECxxx]**

Das Literaturkürzel [SYSTECxxx] verweist Sie auf andere Bedienungsanleitungen von Systec. Das Literaturverzeichnis finden Sie in Kap. 4.2.

Hilfe zur grundsätzlichen Programmierung mit LabVIEW: ni.com

# 2 LabVIEW Funktionsbibliothek für Xemo-Steuerungen

## 2.1 Allgemeines

Programmiert wird eine Xemo-Steuerung mit der Programmiersprache MotionBasic. Die Programmierung kann direkt über die mitgelieferte Entwicklungsumgebung IDE erfolgen oder über eine Hochsprache. Für letztere steht die Xemo-DLL zur Verfügung. Die Xemo-DLL ist auch die Grundlage für die Programmierung mit LabVIEW.
Die Funktionen der Xemo-DLL stehen Ihnen als Sub-VI´s zur Verfügung und sind in einer VI-Bibliothek (*.llb) zusammengefasst. Für die Erstellung der VI-Bibliothek wurde die Xemo-DLL-Version 2.33 eingesetzt und LabVIEW in der Version 2015.
Bis auf wenige Ausnahmen sind alle Funktionen, die im Xemo-DLL-Handbuch [SYSTEC591] in der C-Syntax beschrieben sind, verfügbar. Eine Übersicht der verfügbaren Funktionen finden Sie in Kap. 4.1 ab Seite 21.

**Abgrenzung der Xemo-DLL**

Bei der Systec Xemo-DLL handelt es sich um eine unverwaltete Windows DLL, die unter den folgenden Betriebssystemen eingesetzt werden kann:

| | |
|---|---|
| Windows 95 / 98 | Windows NT 4.0, 2000, XP |
| Windows 7 (32- und 64-Bit) | Windows 8 (32- und 64-Bit) |
| Windows 10 (32- und 64-Bit) | |

## 2.2 Installationshinweise für die LabVIEW Funktionsbibliothek zur Xemo-DLL

**Dateien**

Zur Programmierung mit LabVIEW benötigen Sie folgende Dateien:

| 32-bit | 64-bit | |
|---|---|---|
| XemoDll.llb | Xemo64.llb | LabVIEW VI Bibliothek |
| XemoDll.dll | Xemo64.dll | Xemo-DLL für Windows |
| FTD2XX.dll | FTD2XX64.dll | DLL der USB Schnittstelle (wird von XemoDll benötigt) |

**32-bit-DLLs**

Die 32-bit-.dll-Dateien erhalten Sie mit der Installation der MotionBasic IDE.

**64-bit-DLLs**

Die 64-bit-.dll-Dateien finden Sie auf Ihrer Systec-CD.

Die MotionBasic-Installationsdatei MotionBasicIde_6.5.6.exe finden Sie auf Ihrer Systec-CD im Unterordner DriveSet_Software bzw. Xemo\Xemo_Software.

**Rechte**

Für die Installation der Dateien auf Ihrem PC benötigen Sie die entsprechenden Administrationsrechte.

### 2.2.1 Installation des USB-Treiber

**USB-Treiber**

Zur Verbindung der Xemo-Steuerung mit Ihrem PC über die USB-Schnittstelle benötigen Sie einen eigenen Treiber. Dieser wird automatisch bei der Installation von MotionBasic mitinstalliert.
Prüfen Sie die Installation des USB-Treibers, indem Sie versuchen, mit Xemo!Go auf die Xemo-Steuerung zuzugreifen.

### 2.2.2 Installation der Dateien auf dem PC

Um die Funktionen der Xemo-DLL nutzen zu können, benötigen Sie einen PC mit installiertem LabVIEW. Definitiv geht es mit der Version 2015, weil die VI´s damit erstellt wurden. Die VI´s sind abwärtskompatibel bis zur LabVIEW Version 8.0.

Je nachdem, ob Sie ein 32- oder 64-bit-LabVIEW auf einem 32- oder 64-bit-Rechner in Kombination mit einer Xemo mit oder ohne Ethernet-Schnittstelle verwenden, müssen Sie unterschiedliche DLLs in den Windows-Systemordner bzw. das LabVIEW-Programmverzeichnis kopieren.

Die folgende Tabelle soll Ihnen helfen, die je passenden DLLs für die unterschiedlichen Konfigurationen in die richtigen Zielordner zu kopieren.

<table>
<tr><th></th><th>32-bit-Rechner</th><th>64-bit-Rechner</th><th>VI-Bibliothek</th></tr>
<tr><th>Zielordner<br>Konfiguration</th><td>{WINDOWS}\system32</td><td>{WINDOWS}\SysWOW64</td><td>{LabVIEW-Programmverzeichnis}\user.lib_express\</td></tr>
<tr><td>32 bit LabVIEW</td><td>XemoDLL.dll</td><td>XemoDLL.dll</td><td rowspan="3">XemoDll.llb</td></tr>
<tr><td>Xemo ohne Ethernet</td><td>FTD2XX.dll</td><td>FTD2XX64.dll</td></tr>
<tr><td>Xemo mit Ethernet</td><td>-</td><td>-</td></tr>
<tr><td>64 bit LabVIEW</td><td rowspan="3">-</td><td>Xemo64.dll</td><td rowspan="3">Xemo64.llb</td></tr>
<tr><td>Xemo ohne Ethernet</td><td>FTD2XX64.dll</td></tr>
<tr><td>Xemo mit Ethernet</td><td>-</td></tr>
</table>

**Xemo DLL**

**32-bit-LabVIEW**

Für das 32-bit-LabVIEW brauchen Sie die Datei XemoDLL.dll ab der Version 2.18. Diese finden Sie im Verzeichnis {Programme}/Systec/MotionBasic_6.5.6.

**64-bit-LabVIEW**

Für das 64-bit-LabVIEW brauchen Sie die Datei Xemo64.dll. Diese finden Sie auf Ihrer Systec-CD im Ordner DriveSet_Software\Weitere_-Programmiersprachen\XemoDLL\64bit bzw. Xemo_Software\Weitere_Programmiersprachen\XemoDLL\64bit.

**USB-Treiber**

**Xemo ohne Ethernet** Die Datei FTD2XX.dll für einen 32-bit-Rechner finden Sie im Verzeichnis {Programme}/Systec/ MotionBasic_6.5.6.

Für einen 64-bit-Rechner wird die Datei FTD2XX64.dll benötigt, welche bei der Installation von MotionBasic im Ordner {WINDOWS}\system32\DriverStore\ftdibus.inf_amd64_4f6d99ceb69de87e \amd64 abgelegt wird.

**Xemo mit Ethernet** Die Xemo-Steuerungen mit integrierter Ethernet-Schnittstelle nutzen einen allgemeinen Windows-Treiber. Es ist kein zusätzliches Kopieren von DLLs notwendig.

**VI-Bibliothek** Die XemoDll.llb und die Xemo64.llb finden Sie auf Ihrer Systec-CD im Ordner DriveSet_Software\Weitere_Programmiersprachen\LabVIEW VI bzw. Xemo\Xemo_Software\Weitere_Programmiersprachen\LabVIEW VI.

### 2.2.3 Installation in LabVIEW

Um die Funktionen der Xemo-Dll in die Funktionspalette von LabVIEW aufnehmen zu können, muss die VI-Bibliothek in das Menü aufgenommen werden. Dazu klicken Sie in einem beliebigen LabVIEW Blockdiagramm oder in einem Frontpanel in der Menüleiste „Werkzeuge“ unter „Fortgeschritten“ auf den Menüpunkt „Palette bearbeiten …“.
In der Funktionen-Palette (nicht Bedienelemente) wählen Sie „Eigene Bibliotheken“ aus. Im anschließend auftauchenden (standardmäßig leeren) Untermenü öffnen Sie durch Rechtsklick das Untermenü „Einfügen“ und „Unterpalette…“. Dort selektieren Sie den Punkt 3 - „Mit LLB verknüpfen(.llb)“ - aus. Dann wird ein Fenster angezeigt, in dem Sie den Pfad suchen, in dem sich die XemoDll.llb- bzw. die Xemo64.llb -Datei, die mitgeliefert wurde, befindet. Im Fenster „Elemente- und Funktionen-Palette bearbeiten“ wählen Sie abschließend noch den Punkt „Änderungen Speichern“ aus. Nun sollte in der Funktionen-Palette unter „Eigene Bibliotheken“ der Menüpunkt „Xemo“ angezeigt werden, unter dem die Sub-VIs mit den DLL-Funktionen aufgelistet sind.

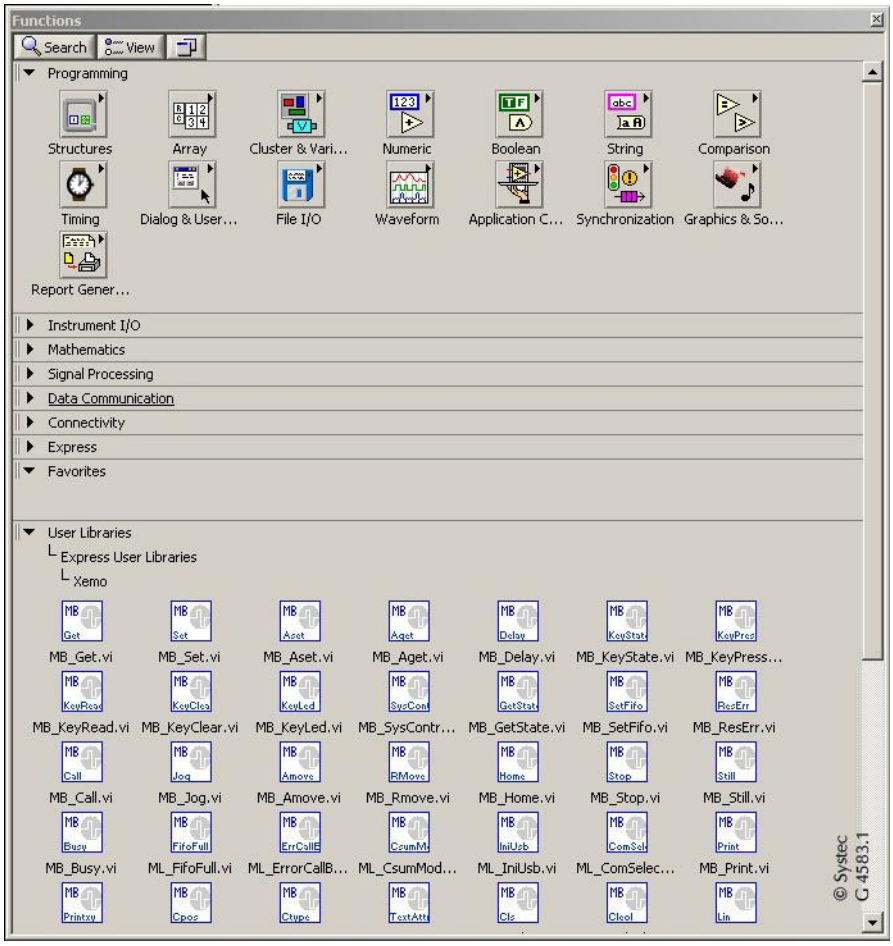


**Abb. 1 In der Funktionen-Palette werden unter „User Libraries“ die Xemo-DLL-Befehle dargestellt**

**Test der Version**

Da Sie testen sollten, ob die aktuelle Version der DLL verwendet wird, empfiehlt es sich, als erstes einmal das VI „dll-Version“ aufzurufen und mit einem Anzeigeelement zu verbinden. Nachdem Sie das Ganze einmal haben laufen lassen, sollte in der Anzeige die Version der aktuell verwendeten Xemo-DLL stehen, z.B. 2.33.

## 2.3 Genereller Aufbau eines Systec VIs

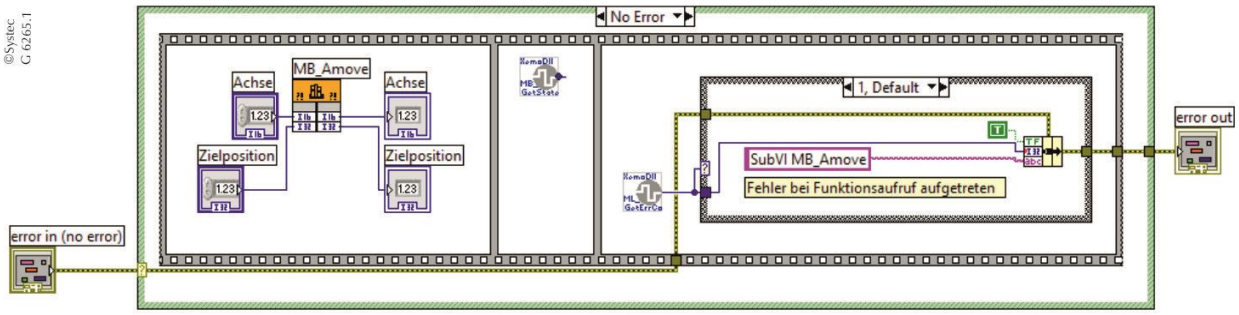


**Abb. 2 Innerer Aufbau eines Systec LabVIEW VIs anhand des Amove-Kommandos**

Jedes Kommando ist eingebettet in eine Status- und eine Fehlerabfrage.

Im linken Teil ist das eigentliche Kommando mit seinen Übergabeschnittstellen dargestellt.
Für die Achsnummer und die Zielposition werden Zahlen erwartet, ebenso werden Zahlen an die Xemo-Steuerung weitergegeben.

In der Mitte sehen Sie die Statusabfrage, welche z.B. den Status des FIFOs überwacht.

Rechts ist das Verhalten des VIs in verschiedenen Situationen definiert.

# 3 Programmierung

## 3.1 Allgemeines

Programmiert wird eine Xemo-Steuerung mit der Programmiersprache MotionBasic. Die Programmierung kann direkt über die mitgelieferte Entwicklungsumgebung IDE erfolgen oder über eine Hochsprache. Für letztere steht die Xemo-DLL zur Verfügung. Die Xemo-DLL ist die Grundlage für die VI-Bibliothek, d.h. alle DLL-Befehle bzw. Funktionen stehen Ihnen als Sub-VI´s zur Verfügung.

Die Darstellung eines Befehls in MotionBasic, mit der Xemo DLL und als Sub-VI wird im Folgenden am Beispiel des Setzens eines Systemparameters gezeigt.

**Tipp**

Die Eigenschaften eines Positioniersystems wie Anzahl der Achsen, zulässige Verfahrgeschwindigkeiten, Leistungsdaten der Motoren etc. werden in der Xemo-Steuerung durch Systemparameter festgehalten. MotionBasic kann die Systemparameter lesen und setzen. Jeder Systemparameter hat eine Registernummer und einen Bezeichner. Durch Wertzuweisungen werden die Systemparameter eingestellt.

Die Befehle für das Setzen eines Systemparameters lauten:

Anweisung in MotionBasic: Set (Parameter, Wert)

Anweisung der Xemo DLL: MB_Set (Parameter, Wert)

Im Beispiel wird die Baudrate des CAN-Busses eingestellt. Dazu dient der Systemparameter mit der Nummer „1034“ und dem Bezeichner „_CanMode“. Über eine Wertzuweisung erfolgt die Auswahl der Baudrate. Mit dem Wert 2 wird eine Baudrate von 500 Kbit/s eingestellt.

Anweisung in MotionBasic: Set (1034, 2)

Anweisung der Xemo DLL: MB_Set (1034, 2)

Sub-VI in LabVIEW:

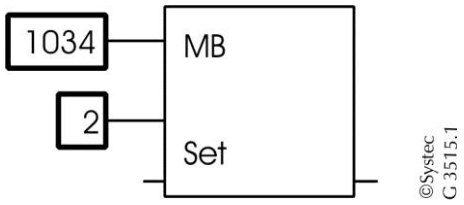


**Abb. 3 Xemo-DLL-Funktion als Sub-VI dargestellt**

Die Zuweisung der Systemparameternummer und des zugehörigen Wertes erfolgt in LabVIEW über Eingangsfelder. In dieser Art werden alle Befehle der Xemo-DLL dargestellt.

Detaillierte Informationen zu Systemparametern finden Sie im achten Kapitel der MotionBasic-Programmieranleitung [SYSTEC717]. Die Xemo-DLL Befehle sind im Xemo Windows DLL Anwenderhandbuch [SYSTEC591] beschrieben.

**Hinweis**

Beachten Sie auch die Nummerierung der Achsen und die Einheit der Werte, z.B. geben Sie den Parameter _Speed in Benutzereinheiten an.

## 3.2 Anweisungen, die jedes Programm beinhalten muss

Mit der Bezeichnung „ML“ werden Funktionen beschrieben, die Aufgaben innerhalb der DLL übernehmen.

**ML_IniCom**
**ML_IniUSB**
**ML_IniTCP**

Vor dem ersten Aufruf eines MotionBasic-Kommandos muss eine Kommunikationschnittstelle geöffnet werden. Die Kommandos ML_IniCom, ML_IniUSB und ML_IniTCP öffnen eine Kommunikationsschnittstelle.

**Hinweis**

Bei Xemo-Steuerungen mit integrierter Ethernet-Schnittstelle muss die Kommunikationsschnittstelle in der Xemo-Steuerung eingestellt werden. Wählen Sie dazu mit der XemoUpdate.exe im Reiter Setup über Com-Device die gewünschte Schnittstelle aus und klicken Sie auf Schreiben – per Betriebsartenschalter in Stellung A und USB-Kabel. Die XemoUpdate.exe wird mit der MotionBasic IDE auf Ihrem PC installiert.

**ML_IniCom**

Über den Parameter ComNo wird entweder die USB-Schnittstelle (ComNo = 0) oder einer der beiden Com-Ports (ComNo = 1 oder 2) ausgewählt. Wird die RS232-Schnittstelle ausgewählt (Com1, Com2), erfolgt über „Baud“ die Angabe der Baudrate. Diese Einstellung entfällt bei Wahl der USB-Schnittstelle.

**ML_IniUSB**

Wollen Sie mehr als eine Xemo-Steuerung per USB-Schnittstelle ansprechen, initialisieren Sie die Kommunikationsschnittstellen über ML_IniUSB. Die Schnittstellen werden so in Abhängigkeit von der Xemo-Seriennummer eindeutig zugewiesen.

**ML_IniTCP**

Wollen Sie die integrierte Ethernetschnittstelle Ihrer Xemo-Steuerung als Kommunikationsschnittstelle nutzen, initialisieren Sie diese über ML_IniTCP mittels der IP- und Port-Adresse der Xemo-Steuerung.

**Hinweis**

Die Portadresse ist derzeit auf einen Wert festgelegt, den Sie der Xemo Windows DLL Anwenderhandbuch [SYSTEC591] entnehmen können.

Die IP-Adresse können Sie mit der XemoUpdate.exe, welche mit der MotionBasic IDE auf Ihrem PC installiert wurde, lesen und schreiben – per Betriebsartenschalter in Stellung A und USB-Kabel.

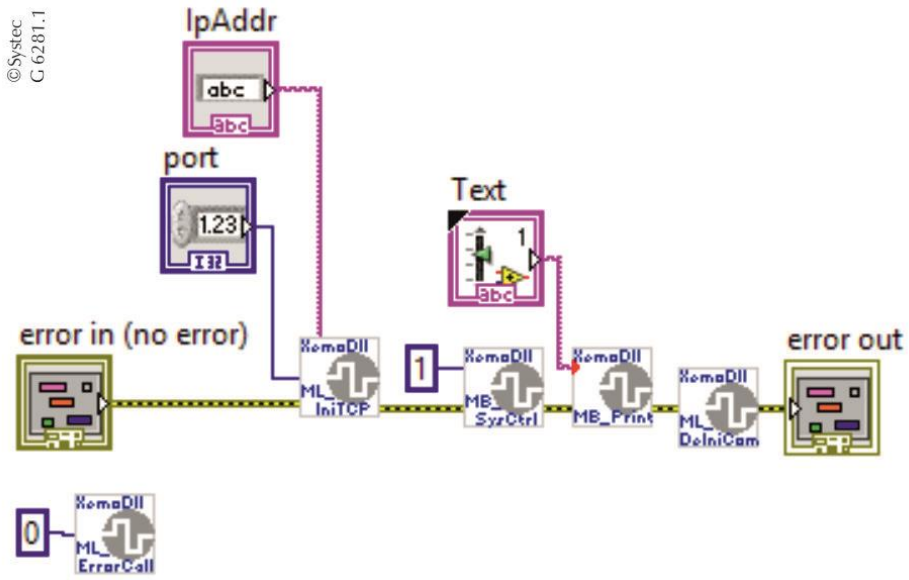


**Abb. 4 Initialisierung der Kommunikationsschnittstelle über ML_IniTCP**

**ML_ErrorCallBack** Mit dieser Routine kann eine applikationsspezifische Call-Back-Routine installiert werden, die im Fehlerfall von der DLL aufgerufen wird. Ist dies unerwünscht, muss direkt nach dem Aufruf von IniCom ML_ErrorCallBack mit dem Parameter 0 aufgerufen werden (0 = keine Fehlerbehandlungsroutine).

**ML_DeInitCom** Vor Programmende müssen in jedem Fall mit der Funktion ML_DeIniCom die Kommunikationsschnittstellen geschlossen werden, sonst kann es zu undefiniertem Verhalten der Steuerung kommen.

## 3.3 Programmbeispiel

### 3.3.1 Einschalten und Referenzfahrt

**Einschaltmoment** Die Xemo-Steuerung weiß nach dem Einschalten nicht, welche Maschine sie steuern muss und darf selbsttätig auch keine Funktion ausführen. Aus diesem Grund werden im Einschaltmoment alle Systemparameter mit von Systec vorgegebenen Werten, den „Default-Werten“, beschrieben. Damit befindet sich die Steuerung in einem definierten Zustand.

**Initialisierung** Nach dem Einschaltmoment muss durch eine entsprechende Programmierung eine Initialisierung erfolgen. Dabei werden die vorgegebenen Werte der Systemparameter mit den spezifischen der jeweiligen Anwendung überschrieben, um anschließend die Referenzfahrt und die eigentlichen Bearbeitungsprogramme ausführen zu können.

Mit LabVIEW haben Sie verschiedene Möglichkeiten der Initialisierung:

#### 3.3.1.1 Initialisierung mit Set/ASet

**Set / Aset**

Sie können jeden Parameter einzeln über seine Nummer mit dem Befehl Set bzw. Aset setzen (wie in Kap. 3.2 beschrieben).

Ein Beispiel für die Parameter, welche Sie bei der Initialisierung eines einfachen 1-Achs-Systems übergeben müssen, finden Sie in [SYSTEC717], Kapitel 2.2.2.
Hinweise und ein Beispiel zum Setzen des Parameters _BrakeOutp zur Bestimmung des Ausgangs für die Bremsansteuerung finden Sie in [SYSTEC625].

**Hinweis**

- ➔ Beachten Sie, die Einheiten konsistent zu Ihrer benötigten Auflösung einzugeben.
- ➔ Beachten Sie, dass das Setzen der Ein- und Ausgänge über IoSet und nicht über Aset erfolgt.

#### 3.3.1.2 Parameter aus Textdatei lesen

Des Weiteren können Sie Ihre Parameter in eine Textdatei schreiben und diese über ein dazwischengeschaltetes LabVIEW VI und Set bzw. Aset an die Steuerung übergeben.

#### 3.3.1.3 Referenzfahrt

**Referenzfahrt**

Nach Initialisierung der Steuerung erfolgt nun die Referenzfahrt über das Xemo VI Home.
Geben Sie dazu die zu referenzierende Achse an.

Nun können Sie Ihre eigentlichen Bearbeitungsprogramme starten.

#### 3.3.1.4 Initialisierung und Referenzfahrt über Call-VIs

**Mitgelieferte Beispielsoftware und Call-VI nutzen**

Als weitere Möglichkeit steht Ihnen die Nutzung der mitgelieferten Beispielsoftware und des Call VIs zur Verfügung.
In einem gemischten Online-Offline-Betrieb speichern Sie die mitgelieferte Beispielsoftware in modifizierter Form in der Steuerung.
Die Initialisierung Ihrer Achsen erfolgt dann automatisch beim Einschalten der Steuerung, die Referenzfahrt rufen Sie über das Call-VI von Ihrer LabVIEW-Oberfläche (s. Abb. 5) aus auf.

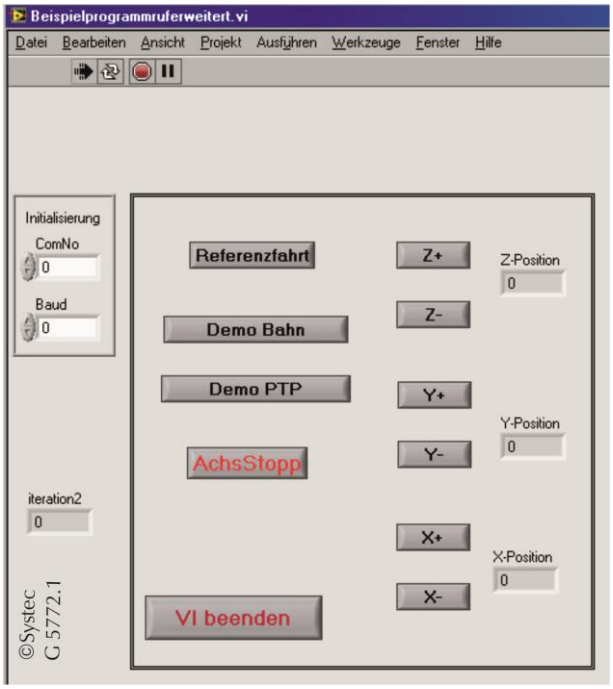


**Abb. 5 Beispiel für eine LabVIEW-Oberfläche: Referenzfahrt und Beispielprogramme können über Schaltflächen aufgerufen werden.**

**Hinweis**

Bei Aufruf der Referenzfahrt muss die Initialisierung der Achsen abgeschlossen sein. Ein Verfahren der Achsen darf erst nach Abschluss der Referenzfahrt erfolgen.

Um die Erfüllung dieser Voraussetzungen sicher zu stellen, können Sie MotionBasic anweisen, nach Abschluss der Initialisierung bzw. der Referenzfahrt einem bestimmten Register einen bestimmten Wert zuzuweisen. Diese Register können vor Ausführung des Call-VIs einfach in LabVIEW durch das Get-VI abgefragt, ihre Werte mit dem Sollwert verglichen und das Ergebnis in einem booleschen Wert ausgegeben werden.

Im Folgenden erfahren Sie, welche Modifikationen Sie an der mitgelieferten Beispielsoftware, einschließlich der Überprüfung der Beendigung von Initialisierung und Referenzfahrt, vornehmen müssen.

**Modifikation der mitgelieferten Beispielsoftware**

Rufen Sie in der MotionBasic IDE die mitgelieferte Beispielsoftware auf und kommentieren Sie in der Main-Datei den Aufruf der Referenzfahrt und des Demoprogramms aus (vgl. Abb. 6).

**DriveSet_main.mb**

```
'*******************************************************************************
' main program of the Driveset software
'*******************************************************************************
sub main
    'initErrorLog                          ' only for the first run, then comment out
    maschine_einschalten_lassen()
    test_hardware()                        ' check, which hardware environment the driveset has
    init()                                 ' call initialization
    set(1,3927)                            ' write "3927" in register 1
    'ref()                                 ' call reference drive
    'demo()                                ' call the demo program
end sub
```



**Abb. 6 Screenshot der Main-Datei: Änderungen gegenüber der mitgelieferten Beispielsoftware sind durch Pfeile gekennzeichnet.**

Fügen Sie set(1,3927) in die Zeile unter die init-Subroutine ein, um dem Register 1 nach erfolgter Initialisierung den Wert 3927 zuzuweisen.

Rufen Sie anschließend den MotionBasic-Code der Subroutine für die Referenzfahrt auf; er befindet sich in der DriveSet_init-Datei.

**DriveSet_init.mb**

```
'*******************************************************************************
' call the homing routines for all axes
'*******************************************************************************
sub @1 ref                                 'labels the sub "ref" as program number 1
dim i, axis as integer

    init_reference()

    if DRIVESET_TYP = COMFORT then
        ref_abfrage()
        cls
        printxy(1,1,"Referenziere....")
        ctype(0)
    endif

    check_ref_sequence()

    for axis = MIN_AXIS to NO_OF_AXES - 1
        i = order(axis) band 7
        if REF_SWITCH(i) >= 0 and REF_SWITCH(i) <= 7 and REF_PORT(i) >= 0 then
            reference_axis_exactly(i)
        else
            reference_axis(i)
        endif
        if (order(axis) < 8) then
            still(_xall)
        endif
    next
    still(_XAll)
    deinit_reference()
    encoder_nachbildung_aktivieren()
    set(2,2416)                                'write "2416" in register 2
end sub
```



**Abb. 7 Screenshot des Referenzfahrt-Aufrufs in der Init-Datei: Der Referenzfahrt-Subroutine ist hier die Programmnummer 1 zugewiesen; nach Beendigung der Referenzfahrt wird Register 2 auf den Wert 2416 gesetzt.**

Im Abschnitt „call the homing routines for all axes“ sehen Sie die Zuweisung der Programmnummer zur Subroutine „ref“ (im Beispiel in Abb. 7 ist es die Programmnummer „1“, erster Pfeil; vgl. auch MotionBasic-Programmieranleitung [SYSTEC717]).

Fügen Sie außerdem set(2,2416) vor end sub ein. Nach Abschluss der Referenzfahrt schreibt MotionBasic so den Wert 2416 in das Register 2.

**error_step.mb**

Um Ihre Fehlerbehandlungsroutinen in LabVIEW durchzuführen, müssen Sie das automatische Einspringen in die OnError-Routine verhindern. Schreiben Sie einen Unterstrich vor den Namen "onerror".

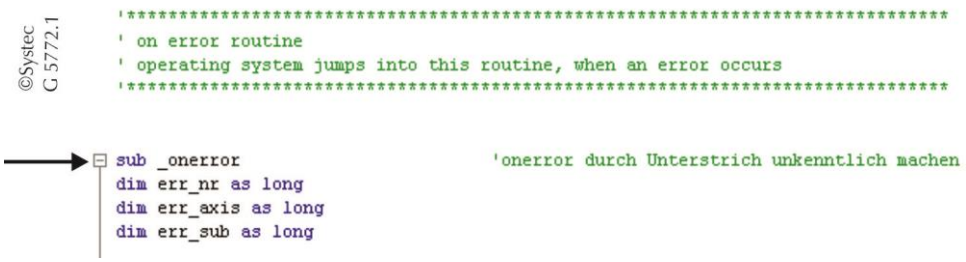


**Abb. 8 OnError-Routine durch Unterstrich für Xemo-Firmware unkenntlich machen.**

Kompilieren Sie Ihr Programm und speichern es in der Steuerung (Menüpunkt „Erstellen in der MotionBasic IDE, näheres hierzu siehe MotionBasic IDE Bedienungsanleitung [SYSTEC875], zur Stellung des Betriebsartenschalters siehe entsprechendes Gerätehandbuch [SYSTEC625] bis [SYSTEC858]).

Fügen Sie im Blockschaltbild Ihrer LabVIEW-Anwendung das Call-VI (s. Abb. 9) z.B. in eine Case-Struktur ein, und geben im Eingangsfeld „ProgNr“ die Programmnummer der Subroutine der Referenzfahrt ein.


**Abb. 9 Das VI „Call“**

Wählen Sie eine boolesche Case-Struktur und verbinden Sie ihren Eingang mit dem Ausgang des Vergleichsoperators. So stellen Sie sicher, dass das Call-VI nur bei gesetzten Registern, also beendeter Initialisierung bzw. Referenzfahrt ausgeführt wird.

**Hinweis**

Beachten Sie, dass Initialisierung und Referenzierung nach einem Reset der Steuerung erneut notwendig sind.

### 3.3.2 Textausgabe auf einem LCD-Display

Im Beispielprogramm wird ein Text auf dem integrierten Display der Xemo R oder Xemo B angezeigt. Bei den anderen Steuerungsmodellen kann ein externes Bedienterminal über den CAN-Bus angesteuert werden.

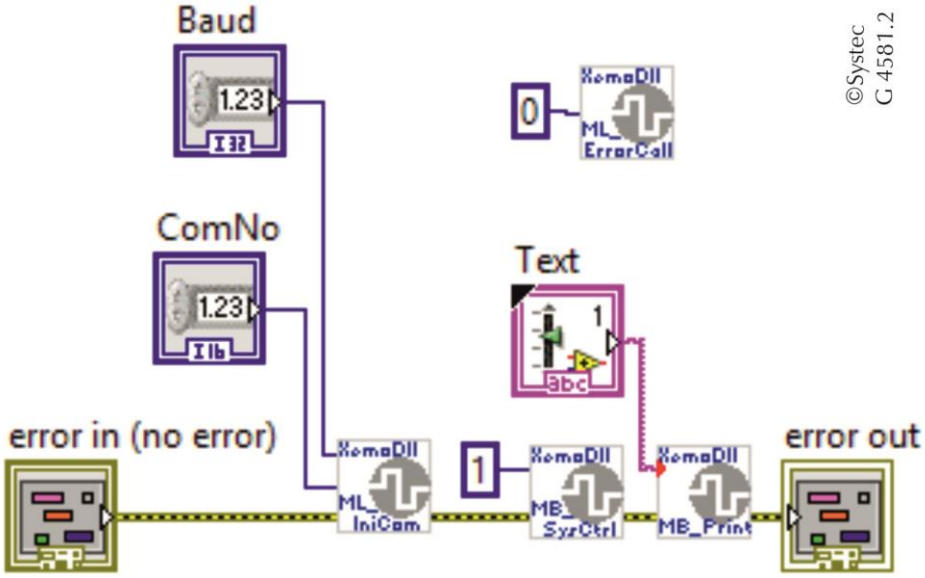


**Abb. 10 Beispielprogramm in Blockdiagramm-Darstellung**

**ML_IniCom**

ML_IniCom öffnet die Kommunikationsschnittstelle. Über den Parameter ComNo wird entweder die USB-Schnittstelle (ComNo = 0) oder einer der beiden Com-Ports (ComNo = 1 oder 2) ausgewählt. Wird die RS232-Schnittstelle ausgewählt (Com1, Com2), erfolgt über „Baud“ die Angabe der Baudrade. Diese Einstellung entfällt bei Wahl der USB-Schnittstelle.

**MB_SysCtrl**

Dieses Kommando bewirkt einen Reset der Steuerung.

**Hinweis**

Wenn Sie im gemischten Betrieb arbeiten, dürfen Sie an dieser Stelle kein MB_SysCtrl ausführen, da die Steuerung durch MB_SysCtrl resettet (neu gestartet) wird und die unter Kap. 3.3.1 ausgeführten Schritte von Initialisierung und Referenzfahrt ausgeführt werden.

**MB_Print**

Mit MB_Print können Texte auf dem LCD-Display dargestellt werden (Xemo R oder Xemo B oder externes Bedienterminal).

**ML_ErrorCallBack**

Mit dieser Routine kann eine applikationsspezifische Call-Back-Routine installiert werden, die im Fehlerfall von der DLL aufgerufen wird. Ist dies unerwünscht, muss direkt nach dem Aufruf von IniCom ML_ErrorCallBack mit dem Parameter 0 aufgerufen werden (0 = keine Fehlerbehandlungsroutine).

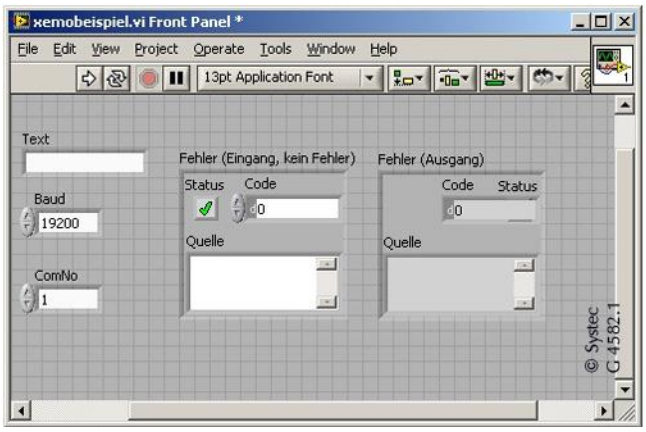


**Abb. 11 Front Panel des Beispielprogramms für die Textausgabe**

Im Front Panel des Beispielprogramms sind die Eingabefelder für die Parameter der Kommunikationsschnittstelle zu sehen. Texte, die in das Textfeld „Text“ eingeben werden, erscheinen auf dem Display der Steuerung.

### 3.3.3 Programmierung von digitalen Ausgängen

Digitale Ausgänge werden durch eine Portadresse und der Bitnummer des jeweiligen Ausgangs angesprochen. Bei der Xemo R/S und B ist das die Portadresse 10 und die Bitnummer 0 bis 7 für die acht Ausgänge (Xemo R/S). Für das Setzen eines Ausgangs erlaubt MotionBasic verschiedene Schreibweisen, die wiederum die Gestaltung des Sub-VI´s beeinflussen.
Im folgenden Vergleich wird ein einzelner Ausgang und zwar Ausgang 4 gesetzt (die Zählweise beginnt bei 0 für Ausgang 1).

Anweisung in MotionBasic: Out (10.3, 1) oder
Out 10.3 = 1

Anweisung der Xemo DLL: MB_Out (10.3, 1)

Sub-VI in LabVIEW:

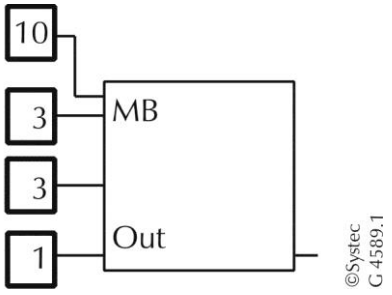


**Abb. 12 Xemo-DLL-Funktion als Sub-VI dargestellt**

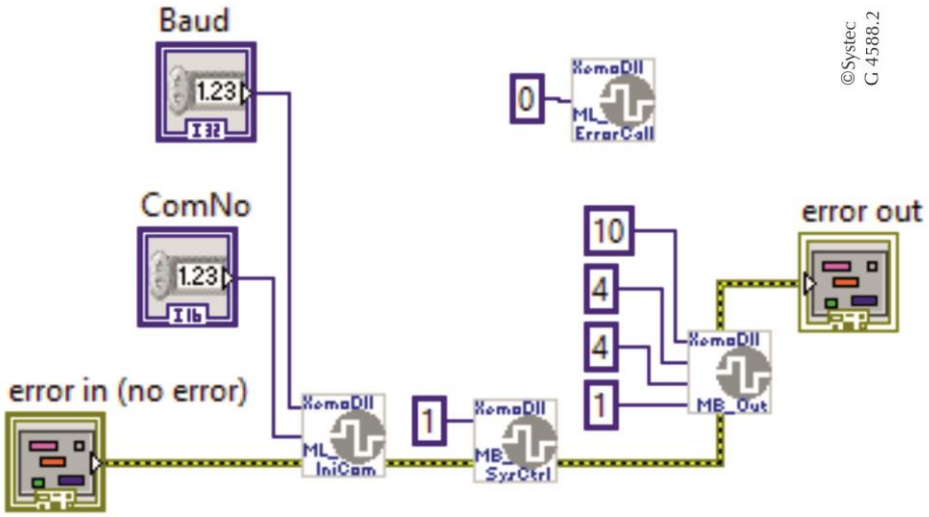


**Abb. 13 Blockdiagramm digitalen Ausgang 5 mit Adresse 10.4 setzen (die Zählweise der Ausgänge beginnt bei 0)**

Im nächsten Beispiel werden mit einer Anweisung mehrere Ausgänge gesetzt. Bei der Portadresse können alle Ausgänge mit der Angabe „0..7“ angesprochen werden. Die Wertzuweisung erfolgt mit einer Dezimal oder Hexadezimalzahl, die steuerungsintern in das Dualsystem umgesetzt wird, um einzelne Ausgänge anzusprechen.
Es sollen die Ausgänge 0, 1 und 4 gesetzt werden:
0001 0011 = 19

Anweisung in MotionBasic: Out 10.0..7 = 19

Anweisung der Xemo DLL: MB_Out (10.0..7, 19)

Sub-VI in LabVIEW:

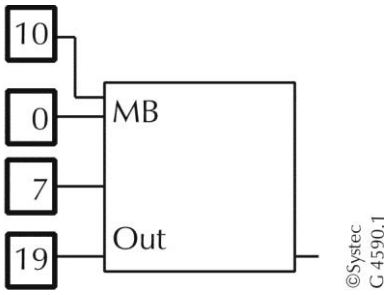


**Abb. 14 Xemo-DLL-Funktion als Sub-VI dargestellt**

### 3.3.4 Programmierung der KeyLeds

Das VI „KeyLed“ enthält neben den üblichen Anschlüssen für die Fehler die Eingangsfelder „on/off“ und „Key“.

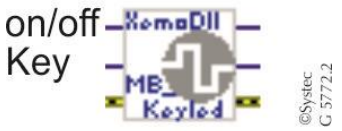


**Abb. 15 Das VI „KeyLed“**

### 3.3.5 Programmierung der Benutzerleuchtdioden bei Xemo R und Xemo S

Die Steuerungsmodelle Xemo R, S, M und Xemo-Step sind mit frei programmierbaren Benutzerleuchtdioden ausgestattet. Die Programmierung erfolgt wie im Kapitel zuvor bei den digitalen Ausgängen beschrieben über die Anweisung „Out“. In diesem Fall ist jedoch die Portadresse 9 gültig. Ansonsten ist die Programmierung identisch.

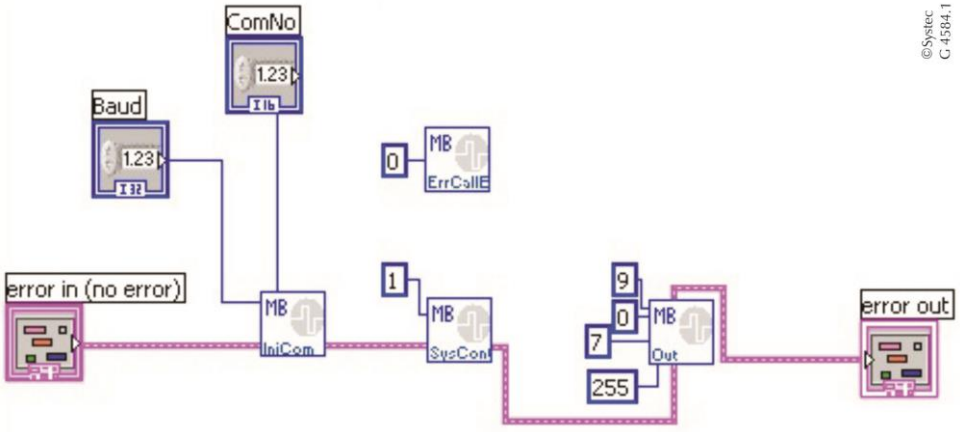


**Abb. 16 Blockdiagramm Benutzerleuchtdioden programmieren; mit der Wertzuweisung 255 werden alle Leuchtdioden eingeschaltet**

# 4 Anhang

## 4.1 Verfügbare DLL-Kommandos

Folgende DLL-Kommandos sind mit der Systec-LabVIEW-Bibliothek verfügbar:

| | DLL-interne Funktionen | |
|---|---|---|
| **Initialisierung** | ML_TimeOut | Timeouts festlegen |
| | ML_IniCom | Öffnen der Kommunikation (COM, USB) |
| | ML_IniUsb | Öffnen der Kommunikation über USB |
| | ML_IniTCP | Öffnen der Kommunikation über Ethernet |
| | ML_ComSelect | Auswahl der Kommunikations-Schnittstelle |
| | ML_DeIniCom | Schließen aller Kommunikations-Schnittstellen |
| | ML_CsumMode | Übertragungsprotokoll mit / ohne Prüfsumme |
| | ML_DllVersion | Version der Xemo-DLL zurückgeben |
| | ML_GetDllVersion | Version der Xemo-DLL ermitteln |
| **Fehlerbehandlung** | ML_ErrorCallBack | Call-Back für allgemeine Fehlerbehandlung |
| | ML_GetErrCode | Fehlernummer abfragen |
| | ML_GetErrState | Fehlerstatus abfragen |
| | ML_SetErrState | Fehlerstatus setzen |
| | ML_LastRunErr | Letzten Laufzeit Fehler der Xemo-Steuerung abfragen |
| | ML_ComErrText | Fehlertext zurückgegeben |
| | ML_GetComErrText | Fehlertext ermitteln |
| **Datenübertragung** | ML_FIFOFull | Testen, ob der Online-FIFO voll ist |
| | ML_PutChar | Ein Byte über die serielle Schnittstelle senden |
| | ML_PutWord | Ein Wort (16 Bit) über die serielle Schnittstelle senden |
| | ML_PutLong | Ein Langwort (32 Bit) über die serielle Schnittstelle senden |
| | ML_GetRcvState | Status der seriellen Schnittstelle abfragen |
| | ML_GetChar | Ein Byte von der seriellen Schnittstelle lesen |
| | ML_GetWord | Ein Wort (16 Bit) von der seriellen Schnittstelle lesen |
| | ML_GetLong | Ein Langwort (32 Bit) von der seriellen Schnittstelle lesen |
| | **MotionBasic-Funktionen** | |
| **Systemkontrolle** | MB_SysCtrl | Anhalten, Abbrechen, Reset, Restart |
| | MB_GetState | Allgemeinen Status abfragen |
| | MB_SetFIFO | Online-FIFO beeinflussen |
| | MB_ResErr | Alle Fehler löschen |
| | MB_Call | Unterprogramm aufrufen |

| | | |
|---|---|---|
| **Systemparameter** | MB_Set | Systemparameter setzen |
| | MB_Seti | Systemparameter setzen |
| | MB_Aset | Achsparameter setzen |
| | MB_Aseti | Achsparameter setzen |
| | MB_IoSet | EA-Parameter setzen |
| | MB_IoSeti | EA-Parameter setzen |
| | MB_Get | Systemparameter lesen |
| | MB_Aget | Achsparameter lesen |
| | MB_IoGet | EA-Parameter lesen |
| **Steuerung einzelner Achsen** | MB_Jog | Geschwindigkeitsmodus |
| | MB_Amove | Absolut positionieren |
| | MB_Rmove | Relativ positionieren |
| | MB_Home | Referenzfahrt |
| | MB_Stop | Stoppen |
| | MB_Still | Stillstand abwarten |
| | MB_Busy | Status einer Achse abfragen |
| **Bahnsteuerung** | MB_Lin, MB_Lin0 | Linearinterpolation mit Eilgeschwindigkeit |
| | MB_Lin1 | Linearinterpolation mit Vorschubgeschwindigkeit |
| | MB_Circle | Kreis mit Radius, Anfangswinkel und Endwinkel |
| | MB_Arc | Kreis-Interpolation mit Radius und Zielposition |
| | MB_Arcc | Kreis-Interpolation mit Mittelpunkt und Zielposition |
| | MB_Arcw | Kreis-Interpolation mit Mittelpunkt und Zielposition |
| **Ein- und Ausgänge** | MB_Out | Ausgänge setzen |
| | MB_Outi | Ausgänge setzen |
| | MB_Sout | Ausgänge synchron setzen |
| | MB_Rout | Ausgänge zurück lesen |
| | MB_In | Eingänge lesen |
| | MB_Waitinp | Auf Eingänge warten |
| | MB_inw | Eingänge wortweise (16 Bit) lesen |
| | MB_Outw | Ausgänge wortweise (16 Bit) setzen |
| | MB_Outwi | Ausgänge wortweise (16 Bit) setzen |
| | MB_Routw | Ausgänge wortweise (16 Bit) zurück lesen |
| **CAN-Bus** | MB_SdoRcv | SDO empfangen |
| | MB_SdoTrm | SDO senden |
| **Textausgabe** | MB_Print | Text an der aktuellen Cursorposition ausgeben |
| | MB_Printxy | Text an der Position x, y ausgeben |
| | MB_Cpos | Cursor positionieren |
| | MB_Ctype | Cursor definieren |
| | MB_TextAttrib | Text-Attribut setzen (Normal, Blinken) |
| | MB_Cls | Bildschirm löschen |

| | | |
|---|---|---|
| | MB_Cleol | Bis zum Ende der Zeile löschen |
| **Tastatureingabe** | MB_Keystate | Status einer Taste |
| | MB_Keypressed | Status: „Taste gedrückt“ |
| | MB_Keyread | Taste lesen |
| | MB_Keyclear | Tastaturbuffer löschen |
| **Terminal** | MB_Keyled | Tasten-LED Ein / Aus |
| **Zeitfunktionen** | MB_Delay | Verweilzeit |
| | MB_Still | Motorstillstand abwarten |
| | MB_Waitinp | Auf Eingänge warten |

Nähere Informationen zu den einzelnen Kommandos finden Sie im Anwenderhandbuch zur Xemo DLL [SYSTEC591] und in der Motion-Basic-Programmieranleitung [SYSTEC717].

## 4.2 Literaturverzeichnis


**[SYSTEC591]** Xemo DLL-Anwenderhandbuch, Doku-Nr. 591-12
© Systec GmbH, Münster 2016

**[SYSTEC625]** Xemo-R/S-Kompaktsteuerung - Bedienungsanleitung, Doku-Nr. 625-12
© Systec GmbH, Münster 2017

**[SYSTEC717]** MotionBasic 6 Programmieranleitung, Doku-Nr. 717-12
© Systec GmbH, Münster 2017

**[SYSTEC735]** Xemo-M-Steuerungsmodul Gerätehandbuch, Doku-Nr. 735-12
© Systec GmbH, Münster 2016

**[SYSTEC764]** Xemo-B-Panelsteuerung Gerätehandbuch, Doku-Nr. 764-12
© Systec GmbH, Münster 2015

**[SYSTEC775]** Xemo!Go-Handbuch, Doku-Nr. 775-12
© Systec GmbH, Münster 2015

**[SYSTEC826]** Strukturierte Fehlersuche, Doku-Nr. 826-12
© Systec GmbH, Münster 2016

**[SYSTEC858]** Xemo-Step Gerätehandbuch, Doku-Nr. 858-12
© Systec GmbH, Münster 2016

**[SYSTEC875]** MotionBasic IDE Bedienungsanleitung, Doku-Nr. 875-12
© Systec GmbH, Münster 2016


Diese Anleitungen finden Sie in Ihrem Handbuchordner, auf der Systec-CD, oder auch zum Herunterladen auf www.systec.de/service/downloads/ .

# 5 Index