DES RS232-Protokoll Windows 32-Bit DLL

HERSTELLERSPEZIFISCHER HINWEIS
Dokumentation DES_WinDLL

CH-Sachseln, 19.08.2004

Die letzte Ausgabe der DES_WinDLL Dokumentation ist auch auf dem Internet unter http://www.maxonmotor.com verfügbar.
(Siehe «Downloads» unter «Service»).

INHALTSVERZEICHNIS

Einleitung

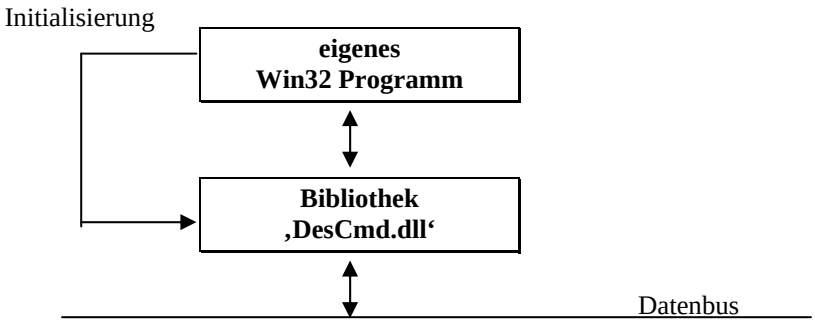
Die Bibliothek DesCmd.dll ist die Implementation des DES RS232-Protokolls für die Anwendung auf einem Personal Computer. Mit dieser Bibliothek kann auf einfache Art und Weise ein eigenes Programm geschrieben werden, um das Verhalten des DES (**D**igitaler **E**C **S**ervoverstärker) zu kontrollieren und zu steuern.

Die Bibliothek ist auf jedem **Windows 32-Bit Betriebssystem** lauffähig und kann in beliebigen Programmiersprachen in den Sourcecode eingebunden werden. Durch das Aufrufen von Bibliotheksfunktionen kann so direkt auf alle DES-Funktionen der DES-Firmware im DES RS232-Modus zugegriffen werden. Der Anwender muss jedoch sicherstellen, dass die Kommunikation richtig konfiguriert ist. Dies geschieht mit einigen Initialisierungsfunktionen am Anfang des Programmstarts.

Die Bibliothek **DesCmd.dll** beinhaltet folgende Funktionen:

- Konfiguration der seriellen Schnittstelle (COM1, ...)
- Zusammenstellen der Datenpakete
- Versenden und lesen dieser Datenpakete
- Fehlerbehandlung

Zu diesem Zweck werden diverse Funktionen der DesCmd.dll Bibliothek verwendet.



Abhängigkeit der Bibliothek

Wie das Inhaltsverzeichnis zeigt, besteht diese Anleitung aus drei Kapiteln und dem Anhang.

- **Kapitel 1:** Alle Bibliotheksfunktionen werden hier beschrieben. Für genauere Angaben steht die Communication Guide zur Verfügung.

- **Kapitel 2:** In diesem Kapitel wird die Frage, wie die Funktionen in die verschiedenen Programmiersprachen eingebunden werden, behandelt. Anhand von einigen Programmierumgebungen wird beschrieben, wie die Funktionen eingebunden werden können. Unter anderem wird in diesem Kapitel auch beschrieben, wie die Funktionen unter 'LabView' benutzt werden können.

- **Kapitel 3:** Das dritte Kapitel zeigt einige Demo-Beispiele von selbstgeschriebenen Anwendungen in verschiedenen Programmiersprachen. Am Anfang jeder Programmiersprache werden die notwendigen Schritte zur Konfiguration der Schnittstelle (COM1, ...) beschrieben. Auch sind hier alle benötigten Dateien notiert, damit das Programm lauffähig ist.
  Die Demo-Beispiele mit der neusten Windows DLL können vom Internet heruntergeladen werden.

- **Anhang:** Im Anhang finden Sie alle Konstantendeklarationen, Strukturdefinitionen und Fehlernummern, welche im Zusammenhang mit der verwendeten Bibliothek auftreten.

1. Bibliothek-Funktionen

1.1. Konfigurationen

DES_InitCommunication

BOOL __stdcall DES_InitCommunication(char portName[], __int32 baudrate, DWORD timeout, DWORD trials);

Beschreibung:	Initialisiert die serielle Schnittstelle (COMx, Baudrate, Timeout, Trials)	
Parameter:	char portName[]:	Angabe der seriellen Schnittstelle. Auswahl von COM1, COM2, ..., COM9
	__int32 baudrate:	Spezifiziert die Baudrate, mit welcher die Kommunikation arbeitet. Mögliche Werte: 9600, 19200, 38400, 57600, 115200 [baud]
	DWORD timeout:	Setzt das Timeout [ms] für Lesefunktionen. Wird die Zeit zwischen zwei aufeinanderfolgenden Bytes grösser als der Wert 'timeout', so bricht die Lesefunktion ab.
	DWORD trials:	Wird das Antwortpacket nicht richtig empfangen, so wird die DES aufgefordert das Antwortpacket nochmals neu zu senden. Der Wert 'trials' gibt an, wo oft dies erfolgen soll.
Rückgabewert:	BOOL:	Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_OpenCommunication

BOOL __stdcall DES_OpenCommunication(char portName[]);

Beschreibung:	Öffnet die serielle Schnittstelle.	
Parameter:	char portName:	Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL:	Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_GetCommunicationSetting

BOOL __stdcall DES_GetCommunicationSetting(char portName[], __int32* baudrate, DWORD* timeout, DWORD* trials);

Beschreibung:	Ermittelt die Einstellungen die an der entsprechenden Schnittstelle vorhanden sind.	
Parameter:	char portName[]:	Serielle Schnittstelle (COM1, ...)
	__int32* baudrate:	Baudrate
	DWORD* timeout:	Timeout der Lesefunktion
	DWORD* trials:	Anzahl Versuche, mit der die DES senden kann.
Rückgabewert:	BOOL:	Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SearchCommunicationSetting

BOOL __stdcall DES_SearchCommunicationSetting(BOOL startAtBeginning, BOOL showMsg, char foundPort[]);

Beschreibung:	Sucht die serielle Schnittstelle (COM1, ...) und deren Einstellungen.	
Parameter:	BOOL startAtBeginning:	TRUE: Beginn der Suche beim ersten Port (Com1,9600) FALSE: Suchbeginn beim nächst möglichen Port
	BOOL showMsg:	TRUE, falls eine Message angezeigt werden soll; FALSE, falls keine Message angezeigt werden soll
	char foundPort[]:	Angabe der gefundenen seriellen Schnittstelle
Rückgabewert:	BOOL:	Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CloseCommunication

BOOL __stdcall DES_CloseCommunication(char portName[]);

Beschreibung:	Schliesst die serielle Schnittstelle und gibt sie für andere Anwendungen frei.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_InitCommunicationDlg

BOOL __stdcall DES_InitCommunicationDlg(char portName[]);

Beschreibung:	Dialog für die Initialisierung der seriellen Schnittstelle (COM1, ...).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.2. Kommando Schicht

1.2.1. Status-Funktionen

DES_ReadSysStatus

BOOL __stdcall DES_ReadSysStatus(char portName[], WORD* sysStatus);

Beschreibung:	Führt den DES RS232 Befehl 'ReadSysStatus' (OpCode = 0x01) aus. Dieser System-Status ist ein 16-bit-Wert mit unterschiedlichen Flags.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD* sysStatus: Zeiger auf die Zustandsvariable (16-bit Status-Variable, siehe im Anhang unter System operating status.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadError

BOOL __stdcall DES_ReadError(char portName[], WORD* error);

Beschreibung:	Führt den DES RS232 Befehl 'ReadError' (OpCode = 0x02) aus. Liefert einen 16-bit-Wert mit verschiedenen Systemfehlern.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD* error: Zeiger auf die Error-Variable (16-bit-Wert mit diversen Error-Meldungen (Anhang: Standard Error Message).
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ClearError

BOOL __stdcall DES_ClearError(char portName[]);

Beschreibung:	Führt den DES RS232 Befehl 'ClearError' (Opcode = 0x03) aus. Löscht alle Fehlernummern aus dem System.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_Reset

BOOL __stdcall DES_Reset(char portName[]);

Beschreibung:	Führt den DES RS232 Befehl 'Reset' (OpCode 0x04) aus. Setzt das ganze System durch einen Software-Neustart zurück.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_Enable

BOOL __stdcall DES_Enable(char portName[], WORD enable);

Beschreibung:	Führt den DES RS232 Befehl 'Enable' (OpCode 0x05) aus. Setzt das System in aktiven oder inaktiven Zustand. Standardmässig ist das System auf Enable. Wird der Hardware-Enable eingeschaltet, so hat dieser Befehl keine Wirkung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD enable: Neuer Zustand des Systems. Mögliche Werte: '0 = Disable' oder '1 = Enable'
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.2.2. Service-Funktionen

BOOL DES_ReadAddrVariable

BOOL DES_ReadAddrVariable(char portName[], WORD address, WORD parType, void* param)

Beschreibung:	Führt den DES RS232 Befehl 'ReadAddrVariable' (OpCode = 0x12) aus. Liest den Wert der angegebenen Adresse aus dem Speicher.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD address: Speicher-Adresse der Variable WORD parType: Angabe des Datentypes. Zur Auswahl stehen '0 = WORD' oder '1 = LWORD' void* param: Zeiger auf den Parameter. Der Wert hat das Format, das unter parType ausgewählt wurde
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.2.3. Systemparameter-Funktionen

DES_ReadTempParam

BOOL __stdcall DES_ReadTempParam(char portName[], WORD parNb, WORD parType, void* param);

Beschreibung:	Führt den DES RS232 Befehl 'ReadTempParam' (OpCode = 0x14) aus. Liest den gewünschten temporären System-Parameter vom DES-RAM.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD parNb: Nummer des System-Parameters; siehe im Anhang unter DES System Parameter WORD parType: Datenformat des Parameters Mögliche Werte: '0 = WORD' oder '1 = LWORD' void* param: Zeiger auf den Rückgabe-Parameter
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetTempParam

BOOL __stdcall DES_SetTempParam(char portName[], WORD parNb, WORD parType, void* param);

Beschreibung:	Führt den DES RS232 Befehl 'SetTempParam' (OpCode = 0x15) aus. Schreibt einen neuen Wert zu einem temporären System-Parameter.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD parNb: Nummer des System-Parameters; siehe im Anhang unter DES System Parameter WORD parType: Datenformat des Parameters Mögliche Werte: '0 = WORD' oder '1 = LWORD' void* param: Zeiger auf den Parameter
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ResetTempParam

BOOL __stdcall DES_ResetTempParam(char portName[]);

Beschreibung:	Führt den DES RS232 Befehl 'ResetTempParam' (OpCode = 0x16) aus. Die im EEPROM enthaltenen permanenten System-Parameter werden in den temporären Bereich kopiert.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SaveTempParam

BOOL __stdcall DES_SaveTempParam(char portName[]);

Beschreibung:	Führt den DES RS232 Befehl 'SaveTempParam' (OpCode = 0x17) aus. Sichert die temporären Parameter in das EEPROM.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadAllTempParam

BOOL __stdcall DES_ReadAllTempParam(char portName[], DES_SysParam* sysParam);

Beschreibung:	Führt den DES RS232 Befehl 'ReadAllTempParam' (OpCode = 0x18) aus. Liest alle temporären System-Parameter.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) DES_SysParam* sysParam: Zeiger auf die temporären Parameter. Siehe im Anhang unter DES System Parameter
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetAllTempParam

BOOL __stdcall DES_SetAllTempParam(char portName[], DES_SysParam* sysParam);

Beschreibung:	Führt den DES RS232 Befehl 'SetAllTempParam' (OpCode = 0x19) aus. Schreibt alle Werte in den temporäre Speicher. Siehe im Anhang unter DES System Parameter.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) DES_SysParam* sysParam: Die Daten-Struktur enthält die neuen System-Parameter.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadVersion

BOOL __stdcall DES_ReadVersion(char portName[], WORD* softVersion, WORD* hardVersion, WORD* appNb, WORD* appVersion);

Beschreibung:	Führt den DES RS232 Befehl 'ReadVersion' (OpCode = 0x1A) aus. Liest die Anwendungs- und die Hardware-Version der DES.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD* softVersion: Zeiger auf die Angabe der Software-Version WORD* hardVersion: Zeiger auf die Angabe der Hardware-Version WORD* appNb: Zeiger auf die Angabe der Anwendungs-Nummer WORD* appVersion: Zeiger auf die Angabe der Anwendungs-Version
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetDefaultSysParam

BOOL __stdcall DES_SetDefaultSysParam(char portName[])

Beschreibung:	Führt den DES RS232 Befehl 'SysPaSetDefault (OpCode = 0x1B) aus. Setzt alle Parameter in den Ursprungszustand (DES System Parameter). Diese Funktion ist erst ab Software Version 0x1050 verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.2.4. Einstell-Funktionen

DES_SetVelocity

BOOL __stdcall DES_SetVelocity(char portName[], short velocity);

Beschreibung:	Führt den DES RS232 Befehl 'SetVelocity' (OpCode = 0x21) aus. Angabe des neuen Drehzahl-Sollwertes [rpm]. Diese Funktion ist nur im Drehzahlregler-Modus verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) short velocity: Angabe des neuen Drehzahl-Sollwertes [rpm]
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetCurrent

BOOL __stdcall DES_SetCurrent(char portName[], short current);

Beschreibung:	Führt den DES RS232 Befehl 'SetCurrent' (OpCode = 0x22) aus. Gibt den neuen Sollwert des Stromes an. Diese Funktion ist nur im Stromregler-Modus verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) short current: Angabe des neuen Strom-Sollwertes [mA]
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_StopMotion

BOOL __stdcall DES_StopMotion(char portName[]);

Beschreibung:	Führt den DES RS232 Befehl 'SetMotion' (OpCode = 0x23) aus. Dieser Befehl wechselt beim Motor vom Stillstand auf Rotation und umgekehrt. Dieser Befehl ist nur im Drehzahlregler-Modus verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.2.5. Überwachungs-Funktionen

DES_ReadVelocityIsMust

BOOL __stdcall DES_ReadVelocityIsMust(char portName[], WORD type, short* isVelocity, short* mustVelocity);

Beschreibung:	Führt den DES RS232 Befehl 'ReadVelocityIsMust' (OpCode = 0x28) aus. Liest die Ist- und Sollwerte der Drehzahl des Motores heraus. Die unterschiedlichen Typen sind erst ab Software Version 0x1040 und höher in Betrieb.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD type: 0 = Mittelwert; 1 = Effektivwert short* isVelocity: type = 0: mittlere Ist-Drehzahl [rpm] type = 1: effektive Ist-Drehzahl [rpm] short* mustVelocity: type = 0: Soll-Drehzahl des Motores [rpm] type = 1: Soll-Drehzahl des Motores [rpm]
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadCurrentIsMust

BOOL __stdcall DES_ReadCurrentIsMust(char portName[], WORD type, short* isQCurrent, short* isDCurrent, short* mustCurAMp, short* currentAngle);

Beschreibung:	Führt den DES RS232 Befehl 'ReadCurrentIsMust' (OpCode = 0x29) aus. Liest den Ist- und den Soll-Wert des Motorenstromes heraus. Die unterschiedlichen Typen sind erst ab Software Version 0x1040 und höher in Betrieb.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD type: 0 = Mittelwert; 1 = Effektivwert short* isQCurrent: type = 0: Mittelwert des Realanteils des aktuellen Stromes [mA] (⇒ Drehmoment) type = 1: Effektivwert des Realanteils des aktuellen Stromes [mA] (⇒ Drehmoment) short* isDCurrent: type = 0: Mittelwert des Imaginäranteils des aktuellen Stromes (≈ 0) type = 1: Effektivwerte des Imaginäranteils des aktuellen Stromes (≈ 0) short* mustCurAmp: type = 0 oder 1: Zeiger auf die Anzeige der Soll-Amplitude des Stromes [mA] short* currentAngle: type = 0 oder 1: Zeiger auf die relative Rotor-Position in einer Umdrehung [qc]
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.2.6. Aufnahmen-Funktionen

DES_SetupRecorder

BOOL __stdcall DES_SetupRecorder(char portName[], WORD samplePeriod, WORD paramNbAddress);

Beschreibung:	Führt den DES RS232 Befehl 'SetupRecorder' (OpCode = 0x30) aus. Bestimmt die Einstellungen für die Datenaufzeichnungen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD samplePeriod: Abtastzeit (als ein Vielfaches von 0.1 ms; z.B. 124 = 12.4ms). WORD paramNbAddress: Nummer eines System-Parameters; Ist die Nummer grösser als 0x300, ist es eine Speicher-Adresse.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_RecordData

BOOL __stdcall DES_RecordData(char portName[], WORD* data, DWORD bufferLength, DWORD* read, DWORD timeout, WORD jump);

Beschreibung:	Führt den DES RS232 Befehl 'RecordData' (OpCode = 0x31) aus. Startet die Aufnahme der Daten. Die Aufnahme stoppt nach 256 Werten. Sprung (jump) erst ab Version 0x1040 und höher verfügbar. Der Sprung funktioniert nur bei digitaler Konfiguration!
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD* data: Zeiger auf den Datenpuffer DWORD bufferLength: Länge des Datenpuffers DWORD* read: Zeiger auf die gelesenen Daten DWORD timeout: Maximaler Zeit-Abstand der Lesefunktion WORD jump: Stromwert des Sprunges. 0 = kein Sprung
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadNVariables

BOOL __stdcall DES_ReadNVariables(char portName[], WORD nbOfVariables,
WORD* parNumbersAddresses, void* dataVector);

Beschreibung:	Führt den DES RS232 Befehl 'ReadNVariables' (OpCode = 0x32) aus. Liest regelmässig die Werte von diversen Variablen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD nbOfVariables: Anzahl der zu lesenden Variablen. WORD* parNumbersAddresses: Zeiger auf die Nummer des System-Parameters; Ist die Nummer grösser als 0x300, so ist es eine Speicher-Adresse void* dataVector: Zeiger auf die Daten
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.2.7. CAN-Bus-Funktionen

DES_ResetCANError

BOOL __stdcall DES_ResetCANError(char portName[]);

Beschreibung:	Führt den DES RS232 Befehl 'ResetCANError' (OpCode = 0x06) aus. Setzt die CAN Errors zurück. Diese Funktion steht erst ab Software Version 0x1040 zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ResetCAN

BOOL __stdcall DES_ResetCAN(char portName[]);

Beschreibung:	Führt den DES RS232 Befehl 'ResetCAN' (OpCode = 0x07) aus. Setzt die CAN Kommunikation auf den Ausgangszustand zurück. Diese Funktion steht erst ab Software Version 0x1040 zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetModuleID

BOOL __stdcall DES_SetModuleID(char portName[], WORD moduleID);

Beschreibung:	Führt den DES RS232 Befehl 'SetModuleID' (OpCode = 0x39) aus. Bestimmt die CAN-Modul-ID (max. 11bit). Die Modul-ID wird beim Einschalten des Stromes durch die DIP-Schalter bestimmt, sofern vorhanden. Während dem Betrieb kann die Modul-ID mit dem Befehl 'SetModuleID' geändert werden (gehen aber nach jedem Stromausfall verloren!). Die Modul-ID bestimmt die IDs für die SDO-Kommunikation (TxSDO ID = 1408 + Modul-ID; RxSDO ID = 1537 + Modul-ID).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD moduleID: Angabe der Modul ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetTPDOID

BOOL __stdcall DES_SetTPDOID(char portName[], WORD transPDOID);

Beschreibung:	Führt den DES RS232 Befehl 'SetTPDOID' (OpCode 0x3B) aus. Bestimmt die Übertragungs-PDO-ID des CAN (max. 11bit).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD transPDOID: Angabe der ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetRPDOID

BOOL __stdcall DES_SetRPDOID(char portName[], WORD receivePDOID);

Beschreibung:	Führt den DES RS232 Befehl 'SetRPDOID' (OpCode = 0x3C) aus. Bestimmt den Empfangs-PDO-ID des CAN (max. 11bit).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD receivePDOID: Angabe der ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SendCANmsg

BOOL __stdcall DES_SendCANmsg(char portName[], WORD id, WORD dataA, WORD dataB, WORD dataC, WORD dataD);

Beschreibung:	Führt den DES RS232 Befehl 'SendCANmsg' (OpCode = 0x3D) aus. Sendet ein Standard-Nachricht-Befehl (CAN Dataframe).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...). WORD id: Angabe der Modul-Adresse. WORD dataA: DatenBytes 2-1. WORD dataB: DatenBytes 4-3. WORD dataC: DatenBytes 6-5. WORD dataD: DatenBytes 8-7.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadModuleID

BOOL __stdcall DES_ReadModuleID(char portName[], WORD* moduleID);

Beschreibung:	Führt den DES RS232 Befehl 'ReadModuleID' (OpCode = 0x3E) aus. Liest die Modul-ID des DES.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD* moduleID: Zeiger auf die Modul-ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetCAN_BCR1_BCR2

BOOL __stdcall DES_ SetCAN_BCR1_BCR2(char portName[], WORD bcr1, WORD bcr2);

Beschreibung:	Führt den DES RS232 Befehl 'SetCAN_BCR1_BCR2' (OpCode = 0x3F) aus. Bestimmt das Zeitverhalten des Konfigurationsregisters 1 und 2 des CAN (erst verfügbar seit Software Version 0x1040 und höher). Für genauer Informationen siehe im Anhang unter CAN Bit Timing.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD bcr1: Berechneter Wert des BCR1 WORD bcr2: Berechneter Wert des BCR2
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetCAN_Bitrate

BOOL __stdcall DES_SetCAN_Bitrate(char portName[], WORD index);

Beschreibung:	Führt den DES RS232 Befehl 'SetCAN_Bitrate' (OpCode = 0x40) aus. Setzt die CAN Bit Timing Register BCR1 und BCR2 auf vorkonfigurierte Werte (0 = 1Mbit/s, 1 = 800kBit/s, 2 = 500kBit/s, 3 = 250kBit/s, 4 = 125kBit/s, 5 = 50kBit/s, 6 = 20kBit/s, 7 = 10kBit/s)
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD index: Index der einzustellenden Uebertragungsrate.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadCANError

BOOL __stdcall DES_ReadCANError(char portName[], WORD* error);

Beschreibung:	Führt den DES RS232 Befehl 'ReadCANError' (OpCode = 0x43) aus. Liest den 16-Bit-Wert des CAN Error Registers.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD* error: 16-Bit Error-Variable.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_GetRemoteData

BOOL __stdcall DES_GetRemoteData(char portName[], WORD id,BYTE opCode, WORD* param, BYTE nbOfParam, WORD* returnParam,BYTE nbOfReturnParam);

Beschreibung:	Führt den DES RS232 Befehl 'GetRemoteData' (OpCode = 0x44) aus. Liest Daten von einzelnen DES, welche mit dem CANBus verbunden sind.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD id: Adresse der DES BYTE opCode: Code für den gewünschten Befehl WORD* param: Zeiger auf den Parameter BYTE nbOfParam: Anzahl der Parameter WORD* returnParam: Zeiger auf die Rückgabe der Parameter BYTE nbOfReturnParam: Anzahl des Rückgabe-Parameter
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ConfigPDO

BOOL __stdcall DES_ConfigPDO(char portName[], WORD action);

Beschreibung:	Führt den DES RS232 Befehl 'ConfigPDO' (OpCode = 0x45) aus. Schaltet die PDO-Kommunikation ein und aus. Der Zustand der PDO-Kommunikation kann mit dem System Parameter 'CAN Config' (SysParam 41, Bit 14) betrachtet werden. Diese Funktion ist erst ab Software Version 0x1040 und höher verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD action: 0 = PDO-Kommunikation ausgeschaltet 1 = PDO-Kommunikation eingeschaltet
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetRTRID

BOOL __stdcall DES_SetRTRID(char portName[], WORD rtrChannel, WORD rtrID);

Beschreibung:	Führt den DES RS232 Befehl 'SetRTRID' (OpCode = 0x46) aus. Bestimmt die RTR-ID (11 Bit). Es stehen zwei Kanäle zur Verfügung. Die aktuellen IDs können in den Systemparameter 39 und 40 betrachtet werden. Diese Funktion ist erst ab Software Version 0x1040 und höher verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD rtrChannel: 0 = RTR0 (Kanal 0); 1 = RTR1 (Kanal 1) WORD rtrID: RTR ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ConfigRTR

BOOL __stdcall DES_ConfigRTR(char portName[], WORD rtrChannel, WORD action);

Beschreibung:	Führt den DES RS232 Befehl 'ConfigRTR' (OpCode = 0x47) aus. Schaltet die RTR-Kommunikation ein und aus. Der aktuelle Zustand kann im System Parameter 'CAN Config' (SysParam 41, Bit 13 = RTR0, Bit12 = RTR1) betrachtet werden. Diese Funktion ist erst ab Software Version 0x1040 und höher verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...). WORD rtrChannel: 0 = RTR0; 1 = RTR1 WORD action: 0 = RTR ein; 1 = RTR aus; 2 = reset
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_AddRTRParameter

BOOL __stdcall DES_AddRTRParameter(char portName[], WORD paramSel, WORD param, BOOL* ack);

Beschreibung:	Führt den DES RS232 Befehl 'AddRTRParameter' (OpCode = 0x4A) aus. Registriert einen neuen Parameter für das RTR. Setzten Sie die RTR Parameter für diesen Kanal zuerst zurück, bevor neue zugeteilt werden. Maximal sind 4 Words (4 x 16 Bit) möglich. Ist der Datespeicher voll, so wird eine negative Bestätigung ('F' = 0x0046) zurückgeschickt.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD paramSel: Bit0: 0 = Kanal 0, 1 = Kanal 1 Bit4: 0 = Parameter Modus, 1 = Adress Modus Bit8: 0 = Word (16 Bit), 1 = LWord (32 Bit) WORD param: Nummer (Parameter Modus) oder Adresse (Adress Modus) des System Parameters BOOL* ack: 0 = Parameter zugefügt, 1 = Parameter nicht zugefügt (Speicher voll)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_GetRTRParameter

BOOL __stdcall DES_GetRTRParameter(char portName[], WORD rtrChannel, WORD index, WORD* ack, WORD* format, WORD* param);

Beschreibung:	Führt den DES RS232 Befehl 'GetRTRParameter' (OpCode = 0x4B) aus. Liest die registrierten RTR Parameter. Maximal sind vier Parameter möglich. Sind weniger als vier Parameter definiert, so wird eine negative Bestätigung ('F' = 0x0046) für die nicht zur Verfügung stehenden Parameter zurückgesendet.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD rtrChannel: 0 = RTR Kanal 0, 1 = RTR Kanal 1 WORD index: 0,1,2,3 (sofern die Parameter zur Verfügung stehen) WORD* ack: 'O' (0x004F) = Parameter verfügbar 'F' (0x0046) = Parameter nicht verfügbar WORD* format: Bit4: 0 = Parameter Modus, 1 = Adress Modus Bit8: 0 = Word (16-bit), 1 = LWord (32-bit) WORD* param: Nummer oder Adresse der registrierten System Parameter (siehe Kapitel SysParam)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.3. Dialog Schicht

1.3.1. Status-Dialoge

DES_ReadSysStatusDlg

BOOL __stdcall DES_ReadSysStatusDlg(char portName[]);

Beschreibung:	Öffnet den Dialog, in dem die Zustände des Systems angezeigt werden.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null

DES_ReadErrorDlg

BOOL __stdcall DES_ReadErrorDlg(char portName[]);

Beschreibung:	Ruft den Dialog für die Fehler-Abfrage auf.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_EnableDlg

BOOL __stdcall DES_EnableDlg(char portName[]);

Beschreibung:	Öffnet den Dialog, in welchem der DES aktiviert oder deaktiviert werden kann.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.3.2. Systemparameter-Dialoge

DES_EditTempParamDlg

BOOL __stdcall DES_EditTempParamDlg(char portName[]);

Beschreibung:	Aufruf des Dialoges, in welchem eine beliebige Variable betrachtet werden kann. Zur Auswahl stehen die 'System Parameter' oder die 'Status Variablen'.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_EditAllTempParamDlg

BOOL __stdcall DES_EditAllTempParamDlg(char portName[]);

Beschreibung:	Dialog, in welchem alle System Parameter betrachtet werden können.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadVersionDlg

BOOL __stdcall DES_ReadVersionDlg(char portName[]);

Beschreibung:	Dialog zum lesen der Hardware- und Software-Version. Angegeben wird noch die Anwendungs-Nummer und die Anwendungs-Version.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...).
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.3.3. Einstell-Dialoge

DES_SetVelocityDlg

BOOL __stdcall DES_SetVelocityDlg(char portName[]);

Beschreibung:	Dialog zur Sollwert-Eingabe der Anzahl Umdrehungen des Motores.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetCurrentDlg

BOOL __stdcall DES_SetCurrentDlg(char portName []);

Beschreibung:	Dialog zur Eingabe, wie hoch der Strom (mA) des Motores sein soll (Sollwert).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.3.4. Überwachungs-Dialoge

DES_ReadVelocityIsMustDlg

BOOL __stdcall DES_ReadVelocityIsMustDlg(char portName[]);

Beschreibung:	Dialog zur Anzeige der Ist- und Sollwerte der Rotorumdrehungen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadCurrentIsMustDlg

BOOL __stdcall DES_ReadCurrentIsMustDlg(char portName[]);

Beschreibung:	Dialog zur Anzeige der Ist- und Sollwerte des Motorenstromes.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.3.5. Aufnahme-Dialoge

DES_SetupRecorderDlg

BOOL __stdcall DES_SetupRecorderDlg(char portName[]);

Beschreibung:	Dialog zum Einstellen der Abtastzeit und der Variable.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_RecordDataDlg

BOOL __stdcall DES_RecordDataDlg(char portName[]);

Beschreibung:	Dialog, zum Einstellen der Aufnahmebedingungen (Abtastzeit, Variable).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadNVariablesDlg

BOOL __stdcall DES_ReadNVariablerDlg(char portName[]);

Beschreibung:	In diesem Dialog können einzelne Variablen ausgewählt werden, um deren Werte anzuzeigen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.3.6. CAN-Bus-Dialoge

DES_ResetCANErrorDlg

BOOL __stdcall DES_ResetCANErrorDlg(char portName[]);

Beschreibung:	Dialog, um die CAN-Fehler zurückzusetzen. Diese Funktion ist erst ab Software Version 0x1040 zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ResetCANDlg

BOOL __stdcall DES_ResetCANDlg(char portName[]);

Beschreibung:	Dialog, um den CAN-Kommunikation in den Ausgangszustand zu setzen. Diese Funktion ist erst ab Software Version 0x1040 zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetModuleIDDlg

BOOL __stdcall DES_SetModuleIDDlg(char portName[]);

Beschreibung:	Dialog, um einem Modul eine bestimmte ID zuzuweisen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetTPDOIDDlg

BOOL __stdcall DES_SetTPDOIDDlg(char portName[]);

Beschreibung:	Dialog, um dem CAN-Bus-Teilnehmer die Übertragungs-PDO-ID zuzuweisen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetRPDOIDDlg

BOOL __stdcall DES_SetRPDOIDDlg(char portName[]);

Beschreibung:	Dialog, um dem CAN-Bus-Teilnehmer den Empfangs-PDO-ID zuzuweisen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SendCANmsgDlg

BOOL __stdcall DES_SendCANmsgDlg(char portName[]);

Beschreibung:	Dialog, um einen Standard-Rahmen von einem CAN-Kommando zu verschicken. Es beinhaltet die ID und Daten von max. vier mal vier Bit.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadModuleIDDlg

BOOL __stdcall DES_ReadModuleIDDlg(char portName[]);

Beschreibung:	Dialog zur Anzeige der Modul-ID.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetCAN_BCR1_BCR2Dlg

BOOL __stdcall DES_SetCAN_BCR1_BCR2Dlg (char portName[]);

Beschreibung:	Dialog für die Angaben der Konfigurationsregister BCR1 und BCR2. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetCAN_BitrateDlg

BOOL __stdcall DES_SetCAN_BitrateDlg(char portName[]);

Beschreibung:	Dialog für die Auswahl der Uebertragungsrate für den CAN Bus. Diese Funktion steht erst ab Software Version 0x1050 zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ReadCANErrorDlg

BOOL __stdcall DES_ReadCANErrorDlg(char portName[]);

Beschreibung:	Dialog zur Anzeige des CAN Errors der an der RS232 angeschlossenen DES.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_GetRemoteDataDlg

BOOL __stdcall DES_GetRemoteDataDlg(char portName[]);

Beschreibung:	Dialog zur Anzeige von Daten einer DES, die auf dem CAN-Bus liegt.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ConfigPDODlg

BOOL __stdcall DES_ConfigPDODlg(char portName[]);

Beschreibung:	Dialog um die PDO-Kommunikation zu konfigurieren. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetRTRIDDlg

BOOL __stdcall DES_SetRTRIDDlg(char portName[]);

Beschreibung:	Dialog um einem RTR Kanal eine ID zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetRTRID0Dlg

BOOL __stdcall DES_SetRTRID0Dlg(char portName[]);

Beschreibung:	Dialog um dem RTR Kanal 0 eine ID zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_SetRTRID1Dlg

BOOL __stdcall DES_SetRTRID1Dlg(char portName[]);

Beschreibung:	Dialog um dem RTR Kanal 1 eine ID zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_ConfigRTRDlg

BOOL __stdcall DES_ConfigRTRDlg(char portName[]);

Beschreibung:	Dialog zum Konfigurieren der RTR Kanäle. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_AddRTRParameterDlg

BOOL __stdcall DES_AddRTRParameterDlg(char portName[]);

Beschreibung:	Dialog um ein RTR Parameter zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_GetRTRParameterDlg

BOOL __stdcall DES_GetRTRParameterDlg(char portName[]);

Beschreibung:	Dialog um einen RTR Parameter zu betrachten. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.4. CAN Kommando Schicht

1.4.1. CAN-Status-Funktionen

DES_CAN_ReadSysStatus

BOOL __stdcall DES_CAN_ReadSysStatus(char portName[], WORD dest, WORD* sysStatus);

Beschreibung:	Führt den DES RS232 Befehl 'ReadSysStatus' (OpCode = 0x01) aus. Dieser System-Status ist ein 16-bit-Wert mit unterschiedlichen Flags (siehe Anhang unter System Operating Status).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD sysStatus: Zeiger auf die Zustandsvariable
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadError

BOOL __stdcall DES_CAN_ReadError(char portName[], WORD dest, WORD* error);

Beschreibung:	Führt den DES RS232 Befehl 'ReadError' (OpCode = 0x02) aus Liefert einen 16-bit-Wert mit verschiedenen Systemfehlern.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD* error: Zeiger auf die Error-Variable. Für genauere Angaben siehe im Anhang unter Standard Error Message.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ClearError

BOOL __stdcall DES_CAN_ClearError(char portName[], WORD dest);

Beschreibung:	Führt den DES RS232 Befehl 'ClearError' (OpCode = 0x03) aus. Löscht alle Fehlernummern aus dem System.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_Reset

BOOL __stdcall DES_CAN_Reset(char portName[], WORD dest);

Beschreibung:	Führt den DES RS232 Befehl 'Reset' (OpCode = 0x04) aus. Setzt das ganze System durch einen Software-Neustart zurück.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_Enable

BOOL __stdcall DES_CAN_Enable(char portName[], WORD dest, WORD enable);

Beschreibung:	Führt den DES RS232 Befehl 'Enable' (OpCode = 0x05) aus. Setzt das System in aktiven oder inaktiven Zustand. Wird der Hardware-Enable eingeschaltet, so hat dieser Befehl keine Wirkung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD enable: Neuer Zustand des Systems: 0 = Disable; 1 = Enable
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.4.2. CAN-Service-Funktionen

DES_CAN_ReadAddrVariable

BOOL __stdcall DES_CAN_ReadAddrVariable(char portName[], WORD dest, WORD address, WORD parType, void* param);

Beschreibung:	Führt den DES RS232 Befehl 'ReadAddrVariable' (OpCode = 0x12) aus. Liest den Wert der angegebenen Adresse aus dem Speicher.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD address: Speicher-Adresse der Variable WORD parType: Angabe des Datentyps: 0 = WORD oder 1 = LWORD void* param: Zeiger auf den Parameter. Der Wert hat den Typ, welcher unter parType ausgewählt wurde
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.4.3. CAN-Systemparameter-Funktionen

DES_CAN_ReadTempParam

BOOL __stdcall DES_CAN_ReadTempParam(char portName[], WORD dest, WORD parNb, WORD parType, void* param);

Beschreibung:	Führt den DES RS232 Befehl 'ReadTempParam' (OpCode = 0x14) aus. Liest den gewünschten temporären System-Parameter vom DES-RAM.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD parNb: Nummer des Systemparameters; siehe im Anhang unter DES System Parameter WORD parType: Datenformat der Variablen: 0 = WORD; 1 = LWORD void* param: Zeiger auf den gewünschten System-Parameter
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetTempParam

BOOL __stdcall DES_CAN_SetTempParam(char portName[], WORD parNb, WORD parType, void* param);

Beschreibung:	Führt den DES RS232 Befehl 'SetTempParam' (OpCode = 0x15) aus. Schreibt einen neuen Wert zu einem temporären System-Parameter.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD parNb: Nummer des System-Parameters; siehe im Anhang unterDES System Parameter WORD parType: Datenformat der Variablen: 0 = WORD; 1 = LWORD void* param: Zeiger auf den System Parameter
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ResetTempParam

BOOL __stdcall DES_CAN_ResetTempParam(char portName[], WORD dest);

Beschreibung:	Führt den DES RS232 Befehl 'ResetTempParam' (OpCode = 0x16) aus. Die im EEPROM enthaltenen permanenten System-Parameter werden in den temporären Bereich kopiert.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SaveTempParam

BOOL __stdcall DES_CAN_SaveTempParam(char portName[], WORD dest);

Beschreibung:	Führt den DES RS232 Befehl 'SaveTempParam' (OpCode = 0x17) aus. Sichert die temporären Parameter in das EEPROM.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadAllTempParam

BOOL __stdcall DES_CAN_ReadAllTempParam(char portName[], WORD dest, DES_SysParam* sysParam);

Beschreibung:	Führt den DES-Bus-Befehl 'ReadAllTempParam' (OpCode = 0x18) aus. List alle temporären System-Parameter.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers DES_SysParam* sysParam: Zeiger auf die temporären Parameter. Siehe im Anhang unter DES System Parameter
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetAllTempParam

BOOL __stdcall DES_CAN_SetAllTempParam(char portName[], WORD dest, DES_SysParam* sysParam);

Beschreibung:	Führt den DES RS232 Befehl 'SetAllTempParam' (OpCode = 0x19) aus. Schreibt alle Werte in den temporären Speicher. Siehe im Anhang unter DES System Parameter.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers DES_SysParam* sysParam: Zeiger auf die Daten-Struktur, die die neuen System-Parameter enthält
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadVersion

BOOL __stdcall DES_CAN_ReadVersion(char portName[], WORD dest, WORD* softVersion, WORD* hardVersion, WORD versionGroup);

Beschreibung:	Führt den DES RS232 Befehl 'ReadVersion' (OpCode = 0x1A) aus. List die Software- und Hardware-Version oder die Applikations-Nummer und die Applikations-Version der DES. Diese Funktion ist erst ab Software Version 0x1040 und höher vorhanden.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers CAN (versionGroup = 0): WORD* softVersion: Zeiger auf Angabe der Software-Version WORD* hardVersion: Zeiger auf Angabe der Hardware-Version CAN (versionGroup = 1): WORD* softVersion: Zeiger auf Angabe der Applikations-Nummer WORD* hardVersion: Zeiger auf Angabe der Applikations-Version WORD versionGroup: 0 = Soft- und Hardware Version 1 = Applikations-Nummer und -Version
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetDefaultSysParam

BOOL __stdcall DES_CAN_SetDefaultSysParam(char portName[])

Beschreibung:	Führt den DES RS232 Befehl 'ReadVersion' (OpCode = 0x1B) aus. Setzt alle Parameter in den Ursprungszustand. Diese Funktion ist erst ab Software Version 0x1050 verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.4.4. CAN-Einstell-Funktionen

DES_CAN_SetVelocity

BOOL __stdcall DES_CAN_SetVelocity(char portName[], WORD dest, short velocity);

Beschreibung:	Führt den DES RS232 Befehl 'SetVelocity' (OpCode = 0x21) aus. Angabe des neuen Drehzahl-Sollwertes [rpm]. Diese Funktion ist nur im Drehzahlregler-Modus verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers short velocity: Angabe des neuen Drehzahl-Sollwertes [rpm]
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetCurrent

BOOL __stdcall DES_CAN_SetCurrent(char portName[], WORD dest, short current);

Beschreibung:	Führt den DES RS232 Befehl 'SetCurrent' (OpCode = 0x22) aus. Gibt den neuen Sollwert des Stromes an. Diese Funktion ist nur im Stromregler-Modus verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers short current: Angabe des neuen Stromwertes [mA]
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_StopMotion

BOOL __stdcall DES_CAN_StopMotion(char portName[], WORD dest);

Beschreibung:	Führt den DES RS232 Befehl 'SetMotion' (OpCode = 0x23) aus. Dieser Befehl wechselt beim Motor zwischen Stillstand und Rotation. Diese Funktion ist nur im Stromregler-Modus verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.4.5. CAN-Überwachungs-Funktionen

DES_CAN_ReadVelocityIsMust

BOOL __stdcall DES_CAN_ReadVelocityIsMust(char portName[], WORD dest, WORD type, short* isVelocity, short* mustVelocity);

Beschreibung:	Führt den DES RS232 Befehl 'ReadVelocityIsMust' (OpCode = 0x28) aus. Liest die Ist- und Soll-Werte der Drehzahl des Motores heraus.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD type: 0 = Mittelwert, 1 = Effektivwert short* isVelocity: type = 0: Mittlere Drehzahl [rpm] type = 1: Effektive Drehzahl [rpm] short* mustVelocity: type = 0 oder 1: Soll-Drehzahl [rpm]
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadCurrentIsMust

BOOL __stdcall DES_CAN_ReadCurrentIsMust(char portName[], WORD dest, WORD type, short* isCurrentQAxis, short* isCurrentDAxis, short* mustCurrentAmp, short* rotorAngle);

Beschreibung:	Führt den DES RS232 Befehl 'ReadCurrentIsMust' (OpCode = 0x29) aus. Liest den Ist- und den Soll-Wert des Motorenstromes heraus.		
Parameter:	char portName[]:	Angabe der seriellen Schnittstelle (COM1, ...)	
	WORD dest:	Adresse des CAN-Bus-Teilnehmers	
	WORD type:	0 = Mittelwert, 1 = Effektivwert	
	short* isCurrentQAxis:	type = 0:	mittlere Realanteil des Stromes (Drehmoment)
		type = 1:	effektiver Realanteil des Stromes (Drehmoment)
	short* isCurrentDAxis:	type = 0:	mittlerer Imaginäranteil des Stromes [mA] (≈ 0)
		type = 1:	effektiver Imaginäranteil des Stromes [mA] (≈ 0)
	short* mustCurrentAmp:	type = 0 oder 1	Strom-Sollwert [mA]
	Short* rotorAngle:	relative Rotor-Position während einer Umdrehung.	
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.		

1.4.6. CAN-Bus-Funktionen

DES_CAN_ResetCANError

BOOL __stdcall DES_CAN_ResetCANError(char portName[], WORD dest);

Beschreibung:	Führt den DES RS232 Befehl 'ResetCANError' (OpCode = 0x06) aus. Setzt die CAN Errors zurück. Diese Funktion ist erst ab Software Version 0x1040 zur Verfügung.	
Parameter:	char portName[]:	Angabe der seriellen Schnittstelle (COM1, ...)
	WORD dest:	Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.	

DES_CAN_ResetCAN

BOOL __stdcall DES_CAN_ResetCAN(char portName[], WORD dest);

Beschreibung:	Führt den DES RS232 Befehl 'ResetCAN' (OpCode = 0x07) aus. Setzt die CAN Kommunikation zurück. Diese Funktion ist erst ab Software Version 0x1040 zur Verfügung.	
Parameter:	char portName[]:	Angabe der seriellen Schnittstelle (COM1, ...)
	WORD dest:	Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.	

DES_CAN_SetModuleID

BOOL __stdcall DES_CAN_SetModuleID(char portName[], WORD dest, WORD moduleID);

Beschreibung:	Führt den DES RS232 Befehl 'SetModuleID' (OpCode = 0x39) aus. Bestimmt die CAN-Modul-ID (max. 11bit). Die Modul-ID wird beim Einschalten des Stromes durch die DIP-Schalter bestimmt, sofern vorhanden. Während dem Betrieb kann die Modul-ID mit dem Befehl 'SetModuleID' geändert werden (geht aber nach jedem Stromeausfall verloren). Die Modul-ID bestimmt die IDs für die SDO-Kommunikation (TxSDO ID = 1408 + Modul-ID; RxSDO ID = 1537 + Modul-ID).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD moduleID: Angabe der Modul-ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetTPDOID

BOOL __stdcall DES_CAN_SetTPDOID(char portName[], WORD dest, WORD transPDOID);

Beschreibung:	Führt den DES RS232 Befehl 'SetTPDOID' (OpCode = 0x3B) aus. Bestimmt die Übertragungs-PDO-ID des CAN (max. 11bit).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD transPDOID: Angabe der ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetRPDOID

BOOL __stdcall DES_CAN_SetRPDOID(char portName[], WORD dest, WORD receivePDOID);

Beschreibung:	Führt den DES RS232 Befehl 'SetRPDOID' (OpCode = 0x3C) aus. Bestimmt den Empfangs-PDO-ID des CAN (max. 11bit).
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD receivePDOID: Angabe der ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadModuleID

BOOL __stdcall DES_CAN_ReadModuleID(char portName[], WORD dest, WORD* moduleID);

Beschreibung:	Führt den DES RS232 Befehl 'ReadModuleID' (OpCode = 0x3E) aus. Liest die Modul-ID der DES.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD* moduleID: Zeiger auf die Modul-ID.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetCAN_BCR1_BCR2

BOOL __stdcall DES_CAN_SetCAN_BCR1_BCR2(char portName[], WORD dest, WORD bcr1, WORD bcr2);

Beschreibung:	Führt den DES RS232 Befehl 'SetCAN_BCR' (OpCode = 0x3F) aus. Bestimmt das Zeitverhalten des CAN-Konfigurationsregisters 1 und 2. Für genauere Angaben siehe im Anhang unter CAN Bit Timing. Funktion erst verfügbar ab Software Version 0x1040 und höher.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD bcr1: Berechneter Wert des BCR1 WORD bcr2: Berechneter Wert des BCR2
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetCAN_Bitrate

BOOL __stdcall DES_CAN_SetCAN_Bitrate(char portName[], WORD dest, WORD index);

Beschreibung:	Führt den DES RS232 Befehl 'SetCANBCR1' (OpCode = 0x40) aus. Setzt die CAN Bit Timing Register BCR1 und BCR2 auf vorkonfigurierte Werte (0 = 1Mbit/s, 1 = 800kBit/s, 2 = 500kBit/s, 3 = 250kBit/s, 4 = 125kBit/s, 5 = 50kBit/s, 6 = 20kBit/s, 7 = 10kBit/s)
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD index: Index der einzustellenden Uebertragungsrate.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ConfigPDO

BOOL __stdcall DES_CAN_ConfigPDO(char portName[], WORD dest, WORD action);

Beschreibung:	Führt den DES RS232 Befehl 'ConfigPDO' (OpCode = 0x45) aus. Schaltet die PDO-Kommunikation ein und aus. Der Zustand der PDO-Kommunikation kann mit dem System Parameter 'CAN Config' (SysParam 41, Bit 14) betrachtet werden. Funktion erst ab Software Version 0x1040 und höher verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD action: 0 = PDO-Kommunikation ausgeschaltet; 1 = PDO-Kommunikation eingeschaltet.
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetRTRID

BOOL __stdcall DES_CAN_SetRTRID(char portName[], WORD dest, WORD rtrChannel, WORD rtrID);

Beschreibung:	Führt den DES RS232 Befehl 'SetRTRID' (OpCode = 0x46) aus. Bestimmt die RTR-ID (11 Bit). Es stehen zwei Kanäle zur Verfügung. Die aktuellen IDs können in den Systemparameter 39 und 40 angesehen werden. Diese Funktion erst ab Software Version 0x1040 und höher verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD rtrChannel: 0 = RTR0 (Kanal 0); 1 = RTR1 (Kanal 1) WORD rtrID: RTR ID
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ConfigRTR

BOOL __stdcall DES_CAN_ConfigRTR(char portName[], WORD dest, WORD rtrChannel, WORD action);

Beschreibung:	Führt den DES RS232 Befehl 'ConfigRTR' (OpCode = 0x47) aus. Schaltet die RTR-Kommunikation ein und aus. Der aktuelle Zustand kann im System Parameter 'CAN Config' (SysParam 41, Bit 13 = RTR0, Bit12 = RTR1) betrachtet werden. Diese Funktion ist erst ab Software Version 0x1040 und höher verfügbar.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD rtrChannel: 0 = RTR0; 1 = RTR1 WORD action: 0 = RTR ein, 1 = RTR aus, 2 = Parameter zurücksetzen
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_AddRTRParameter

BOOL __stdcall DES_CAN_AddRTRParameter(char portName[], WORD dest, WORD paramSel, WORD param, BOOL* ack);

Beschreibung:	Führt den DES RS232 Befehl 'AddRTrParameter' (OpCode = 0x4A) aus. Registriert einen neuen Parameter für das RTR. Setzten Sie die RTR Parameter für diesen Kanal zuerst zurück, bevor neue zugeteilt werden. Maximal sind 4 Words (4 x 16 Bit) möglich. Ist der Datespeicher voll, so wird eine negative Bestätigung ('F' = 0x0046) zurückgeschickt.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD paramSel: Bit0: 0 = Kanal 0, 1 = Kanal 1 Bit4: 0 = Parameter Modus, 1 = Adress Modus Bit8: 0 = Word (16 Bit), 1 = LWord (32 Bit) WORD param: Nummer (Parameter Modus) oder Adresse (Adress Modus) des System Parameters BOOL* ack: 0 = Parameter zugefügt, 1 = Parameter nicht zugefügt (Speicher voll)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_GetRTRParameter

BOOL __stdcall DES_CAN_GetRTRParameter(char portName[], WORD dest, WORD rtrChannel, WORD index, WORD* ack, WORD* format, WORD* param);

Beschreibung:	Führt den DES RS232 Befehl 'GetRTRParameter' (OpCode = 0x4B) aus. Liest die registrierten RTR Parameter. Maximal sind vier Parameter möglich. Sind weniger als vier Parameter definiert, so wird eine negative Bestätigung ('F' = 0x0046) für die nicht zur Verfügung stehenden Parameter gesendet.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers WORD rtrChannel: 0 = RTR Kanal 0, 1 = RTR Kanal 1 WORD index: 0,1,2,3 (insofern die Parameter zur Verfügung stehen) WORD* ack: 'O' (0x004F) = Parameter verfügbar 'F' (0x0046) = Parameter nicht verfügbar WORD* format: Bit4: 0 = Parameter Modus, 1 = Adress Modus Bit8: 0 = WORD (16-bit), 1 = LWORD (32-bit) WORD* param: Nummer oder Adresse der registrierten System Parameter (siehe Kapitel SysParam)
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.5. CAN Dialog-Schicht

1.5.1. CAN-Status-Dialoge

DES_CAN_ReadSysStatusDlg

BOOL __stdcall DES_CAN_ReadSysStatusDlg(char portName[], WORD dest);

Beschreibung:	Öffnet den Dialog, in dem die einzelnen System-Zustände abgebildet werden.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadErrorDlg

BOOL __stdcall DES_CAN_ReadErrorDlg(char portName[], WORD dest);

Beschreibung:	Ruft den Dialog auf, in dem die einzelnen Fehler angezeigt werden.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...). WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_EnableDlg

BOOL __stdcall DES_CAN_EnableDlg(char portName[], WORD dest);

Beschreibung:	Öffnet den Dialog, in dem eingestellt wird, ob ein Bus-Teilnehmer aktiviert oder deaktiviert ist.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.5.2. CAN-Systemparameter-Dialoge

DES_CAN_EditTempParamDlg

BOOL __stdcall DES_CAN_EditTempParamDlg(char portName[], WORD dest);

Beschreibung:	Aufruf des Dialoges, in welchem eine beliebige Variable betrachtet werden kann. Zur Auswahl stehen die 'System Parameter' oder die 'Status Variablen'.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_EditAllTempParamDlg

BOOL __stdcall DES_CAN_EditAllTempParamDlg(char portName[], WORD dest);

Beschreibung:	Dialog, in welchem alle System Parameter betrachtet werden können.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadVersionDlg

BOOL __stdcall DES_CAN_ReadVersionDlg(char portName[], WORD dest);

Beschreibung:	Dialog zum lesen der Hardware- und Software-Version. Angegeben wird noch die Anwendungs-Nummer und die Anwendungs-Version.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...). WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.5.3. CAN-Einstell-Dialoge

DES_CAN_SetVelocityDlg

BOOL __stdcall DES_CAN_SetVelocityDlg(char portName[], WORD dest);

Beschreibung:	Dialog zur Eingabe der Soll-Drehzahl des Motores [rpm].
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetCurrentDlg

BOOL __stdcall DES_CAN_SetCurrentDlg(char portName[], WORD dest);

Beschreibung:	Dialog zur Eingabe des Sollwertes des Stromes [mA].
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.5.4. CAN-Überwachungs-Dialoge

DES_CAN_ReadVelocityIsMustDlg

BOOL __stdcall DES_CAN_ReadVelocityIsMustDlg(char portName[], WORD dest);

Beschreibung:	Dialog zur Anzeige der Ist- und Sollwerte der Rotorumdrehungen [rpm].
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadCurrentIsMustDlg

BOOL __stdcall DES_CAN_ReadCurrentIsMustDlg(char portName[], WORD dest);

Beschreibung:	Dialog zur Anzeige der Ist- und Sollwerte des Motorenstromes [mA].
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

1.5.5. CAN-Bus-Dialoge

DES_CAN_ResetCANErrorDlg

BOOL __stdcall DES_CAN_ResetCANErrorDlg(char portName[], WORD dest);

Beschreibung:	Dialog, um die CAN-Fehler zurückzusetzen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ResetCANDlg

BOOL __stdcall DES_CAN_ResetCANDlg(char portName[], WORD dest);

Beschreibung:	Dialog, um den CAN-Kommunikation zurückzusetzen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetModuleIDDlg

BOOL __stdcall DES_CAN_SetModuleIDDlg(char portName[], WORD dest);

Beschreibung:	Dialog, um einem Modul eine bestimmte ID zuzuweisen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetTPDOIDDlg

BOOL __stdcall DES_CAN_SetTPDOIDDlg(char portName[], WORD dest);

Beschreibung:	Dialog, um einem Modul die Übermittlungs-PDO-ID zuzuweisen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetRPDOIDDlg

BOOL __stdcall DES_CAN_SetRPDOIDDlg(char portName[], WORD dest);

Beschreibung:	Dialog, um einem Modul die Empfangs-PDO-ID zuzuweisen.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ReadModuleIDDlg

BOOL __stdcall DES_CAN_ReadModuleIDDlg(char portName[], WORD dest);

Beschreibung:	Dialog zur Anzeige der Modul ID.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetCAN_BCR1_BCR2Dlg

BOOL __stdcall DES_CAN_SetCAN_BCR1_BCR2Dlg(char portName[], WORD dest);

Beschreibung:	Dialog für die Angaben der Konfigurationsregister BCR1 und BCR2. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetCAN_BitrateDlg

BOOL __stdcall DES_CAN_SetCAN_BitrateDlg(char portName[], WORD dest);

Beschreibung:	Dialog für die Auswahl der Uebertragungsrate für den CAN Bus. Diese Funktion steht erst ab Software Version 0x1050 zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ConfigPDODlg

BOOL __stdcall DES_CAN_ConfigPDODlg(char portName[], WORD dest);

Beschreibung:	Dialog um die PDO-Kommunikation zu konfigurieren. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetRTRIDDlg

BOOL __stdcall DES_CAN_SetRTRIDDlg(char portName[], WORD dest);

Beschreibung:	Dialog um einem RTR Kanal eine ID zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetRTRID0Dlg

BOOL __stdcall DES_CAN_SetRTRID0Dlg(char portName[], WORD dest);

Beschreibung:	Dialog um dem RTR Kanal 0 eine ID zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_SetRTRID1Dlg

BOOL __stdcall DES_CAN_SetRTRID1Dlg(char portName[], WORD dest);

Beschreibung:	Dialog um dem RTR Kanal 1 eine ID zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_ConfigRTRDlg

BOOL __stdcall DES_CAN_ConfigRTRDlg(char portName[], WORD dest);

Beschreibung:	Dialog zum Konfigurieren der RTR Kanäle. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_AddRTRParameterDlg

BOOL __stdcall DES_CAN_AddRTRParameterDlg(char portName[], WORD dest);

Beschreibung:	Dialog um ein RTR Parameter zuzuweisen. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

DES_CAN_GetRTRParameterDlg

BOOL __stdcall DES_CAN_GetRTRParameterDlg(char portName[], WORD dest);

Beschreibung:	Dialog um einen RTR Parameter zu betrachten. Diese Funktion steht ab Software Version 0x1040 und höher zur Verfügung.
Parameter:	char portName[]: Angabe der seriellen Schnittstelle (COM1, ...) WORD dest0: Adresse des CAN-Bus-Teilnehmers
Rückgabewert:	BOOL: Ist die Funktion erfolgreich, so ist der Rückgabewert ungleich null.

2. Einbinden von Bibliotheksfunktionen

Im folgenden Kapitel wird kurz erläutert, wie die verschiedenen Funktionen in eigene Programme eingebunden werden können. Dies ist von Compiler zu Compiler und von Programmiersprache zu Programmiersprache unterschiedlich. Dazu wird in den nächsten Abschnitten anhand einer Auswahl von Programmierumgebungen erläutert, wie die Bibliothek verwendet werden kann.

2.1. Allgemeine Informationen

Damit die Kommunikation richtig funktionieren kann, muss die Bibliothek

DesCmd.dll

eingebunden werden und in einem der Suchverzeichnisse des Systems vorhanden sein.

Mit der Bibliothek DesCmd.dll wird die serielle Schnittstelle (COM1, ...) ausgewählt und konfiguriert. Dieser Vorgang muss direkt beim Öffnen des GUI (Graphisches User Interface) vorgenommen werden.
Am Schluss muss die Schnittstelle wieder freigegeben werden.

Zum Aufrufen der Bibliotheksfunktionen muss die Konvention __**stdcall** verwendet werden. Durch diese Konvention wird geregelt, wie die Parameter auf den Stack gelegt werden und wer verantwortlich ist, den Stack nach der Funktion wieder aufzuräumen!

2.2. Microsoft Visual C++

Um die Bibliotheken in die Programmierumgebung von Microsoft Visual C++ einbinden zu können, werden folgende Dateien benötigt:

- **Def.h**: Konstantendefinitionen und Deklarationen der Bibliotheksfunktionen
- **DesCmd.dll**: 'Data link layer' - Funktionscode
- **DesCmd.lib**: 'Data link layer' - Importbibliothek (COFF Format)

Die folgenden Schritte sind notwendig um die Bibliothek korrekt einzubinden:

Erster Schritt: Es müssen alle Dateien ins Arbeitsverzeichnis des Projektes kopiert werden.

Zweiter Schritt: Die Header-Datei 'Def.h' muss in den Programmcode eingebunden werden. Dies geschieht durch die Anweisung **' #include "Def.h" '.**

Dritter Schritt: Die Datei 'DesCmd.lib' muss zum Projekt hinzugefügt werden. Dazu wählt man den Menüpunkt 'Einstellungen' im Menü 'Projekt'. Unter dem Reiter 'Linker' werden die Datei 'DesCmd.lib' im Feld 'Objekt- / Bibliothek-Module' eingetragen. Die Abbildung 2.1 zeigt diese Einträge.

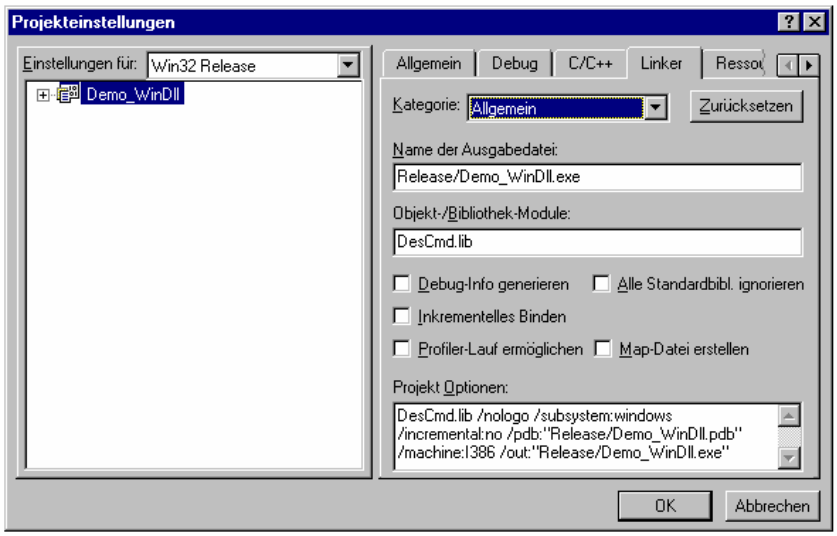


Abbildung 2.1: Projekteinstellungen in Visual C++

Nach diesen drei Schritten kann innerhalb des Programmcodes direkt auf die einzelnen Funktionen zugegriffen werden.

2.3. Microsoft Visual Basic

In der Programmierumgebung von Microsoft Visual Basic werden die folgenden Dateien benötigt, um die Funktionen im eigenen Programm zugänglich zu machen:

- **Def.bas**: Konstantendefinitionen und Deklarationen der Bibliotheksfunktionen
- **DesCmd.dll**: 'Data link layer' - Funktionscode

Die folgenden Schritte sind notwendig um die Bibltiotheken korrekt einzubinden:

Erster Schritt: Alle Dateien müssen in das Verzeichnis des aktuellen Projektes kopiert werden.

Zweiter Schritt: Das Programmodul 'Def.bas' muss eingebunden werden. Dazu wird im Menu 'Projekt' der Menupunkt 'Modul hinzufügen' angewählt. Im Reiter 'Vorhanden' kann darauf die Datei 'Def.bas' ausgewählt und hinzugefügt werden. Die Abbildung 2.2 zeigt diesen Vorgang.

Dritter Schritt: Zusätzlich muss die Datei 'DesCmd.dll' ins Windows Systemverzeichnis oder irgendein anderes sichtbares Verzeichnis kopiert werden.

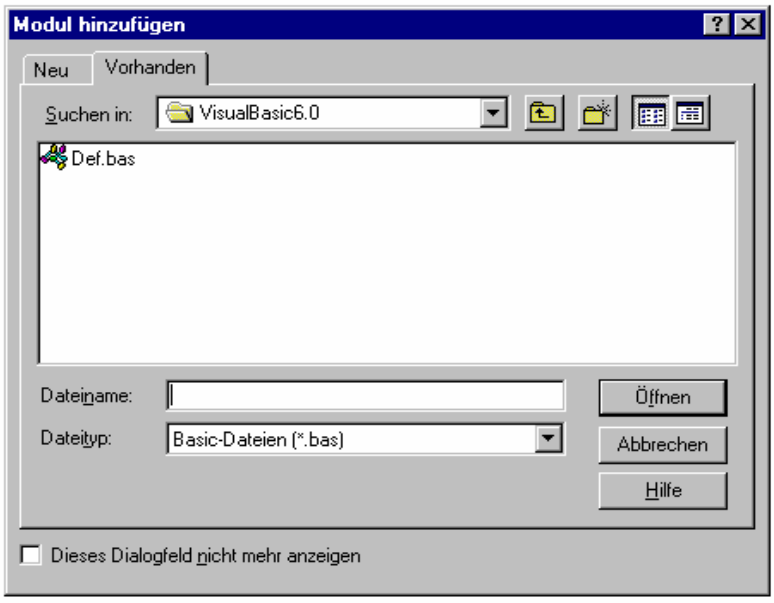


Abbildung 2.2: Modul einfügen in Visual Basic

Ab jetzt können die verschiedenen DES RS232 Befehle direkt vom Programmcode aufgerufen werden wie jede normale Visual Basic Funktion.

Vorsicht!

Bei der Übergabe von Parametern muss aufgepasst werden, dass die Parameterwerte richtig abgebildet werden. Visual Basic kennt keine vorzeichenlose Datentypen. Verschiedene DES RS232 Befehle verlangen jedoch vorzeichenlose Datentypen. Es ist somit darauf zu achten, dass die vorzeichenbehafteten Datentypen richtig auf die vorzeichenlosen Datentypen abgebildet werden. In der Datei 'Def.bas' wurde im Hilfetext darauf hingewiesen, falls solch ein Fall auftreten sollte.

2.4. Borland C++ Builder

Um die Bibliotheksfunktionen in die Programmierumgebung von Borland C++ Builder einzubinden benötigt man folgende Dateien:

- **Def.h** Konstantendefinition und Deklaration der Bibliotheksfunktionen
- **DesCmd.dll** 'Data link layer' - Funktionscode
- **DesCmd.lib** 'Data link layer' - Importbibliothek (OMF Format)

Die Bibliotheks-Datei hat bei Borland ein anderes Format, als bei Microsoft Visual C++. Es muss also darauf geachtet werden, dass die richtige Version verwendet wird.

Das Einbinden der Funktion geschieht in den folgenden Schritten:

Erster Schritt: Die oben aufgeführten Dateien müssen in das Arbeitsverzeichnis des Projektes kopiert werden.

Zweiter Schritt: Durch die Anweisung **' #include "Def.h" '** werden die Konstanten-Definitionen und Funktionsdeklarationen in das Programm eingebunden.

Dritter Schritt: Die Datei 'DesCmd.lib' muss zum Projekt hinzugefügt werden.

Die Funktionen sind nach diesen drei Schritten im eigenen Programmcode aufrufbar.

2.5. Borland Delphi

Um die Bibliotheksfunktionen in die Programmierumgebung von Borland Delphi einbinden zu können, benötigt man die folgenden Dateien:

- **Def.pas**: Konstantendefinitionen und Deklarationen der Bibliotheksfunktionen
- **DesCmd.dll**: 'Data link layer' - Funktionscode

Das Einbinden der Funktionen geschieht in folgenden Schritten:

Erster Schritt: Die oben aufgeführten Dateien müssen in das Arbeitsverzeichnis des Delphi Projektes kopiert werden.

Zweiter Schritt: Durch die Anweisung **'Uses Def'** werden die Konstantendefinitionen und die Funktionsdeklarationen in das Programm eingebunden.

Die Funktionen sind nach diesen zwei Schritten im eigenen Programmcode aufrufbar.

2.6. National Instruments LabView

Um die Bibliotheksfunktionen in LabView einzubinden, braucht es ein wenig mehr Aufwand als bei anderen Programmierumgebungen. LabView stellt einen Funktionsblock zur Verfügung, um externe Bibliotheksfunktionen einbinden zu können. Dabei muss für jede Funktion ein Funktionsblock erstellt werden, der die Definitionen der Funktionsparameter enthält.

Um trotzdem einen problemlosen und schnellen Einstieg in die Programmierung mit LabView zu ermöglichen werden alle DES Befehle bereits als SubVI Blöcke zur Verfügung gestellt. Die Blöcke können so direkt in das eigene Programm kopiert werden. Es wird eine LabView-Bibliothek **(DesCmd.llb)** mitgeliefert, welche verschiedene Funktionsgruppen VIs enthält. Jede Funktionsgruppe enthält eine Anzahl von SubVIs. Jedes SubVI steht dabei für einen DES RS232 Befehl und kann direkt verdrahtet werden.

Die folgenden Dateien und Verzeichnisse müssen vorhanden sein, um die Funktionsblöcke einbinden zu können.

Dateien:

- **DesCmd.dll**: 'Application layer' – Funktionscode
- **DesCmd**.**llb**: LabView Bibliothek: Initialisation.vi (Enthält SubVIs)
  MonitorFunctions.vi (Enthält SubVIs)
  Settings.vi (Enthält SubVIs)
  SystemParameter.vi (Enthält SubVIs)
  Status.vi (Enthält SubVIs)
  Recording.vi (Enthält SubVIs)
  CANFunctions.vi (Enthält SubVIs)
  CANCommands.vi (Enthält SubVIs)

Verzeichnisse:

- **\DesCmd\ Initialisation*.vi** SubVIs von DesCmd.llb; Initialisation.vi
- **\DesCmd\ MonitorFunctions*.vi** SubVIs von DesCmd.llb; MonitorFunctions.vi
- **\DesCmd\ Settings*.vi** SubVIs von DesCmd.llb; Settings.vi
- **\DesCmd\ SystemParameter*.vi** SubVIs von DesCmd.llb; SystemParameter.vi
- **\DesCmd\ Status*.vi** SubVIs von DesCmd.llb; Status.vi
- **\DesCmd\ Recording*.vi** SubVIs von DesCmd.llb; Recording.vi
- **\DesCmd\ CANFunctions*.vi** SubVIs von DesCmd.llb; CANFunctions.vi
- **\DesCmd\ CANCommands*.vi** SubVIs von DesCmd.llb; CANCommands.vi

Um ein SubVI in ein eigenes LabView Programm einbinden zu können, müssen die folgenden Schritte durchgeführt werden.

Erster Schritt: Die aufgeführten Dateien und Verzeichnisse müsssen in ein eigenes Arbeitsverzeichnis kopiert werden.

Zweiter Schritt: Öffnen Sie die Bibliothek 'DesCmd.lib'. Wählen Sie im Dialog 'Dateidialog' die entsprechende Funktionsgruppe. Die Abbildung unten zeigt diesen Dialog. Danach erscheint ein Frontpanel-Fenster. Dieses Fenster enthält keine Elemente.

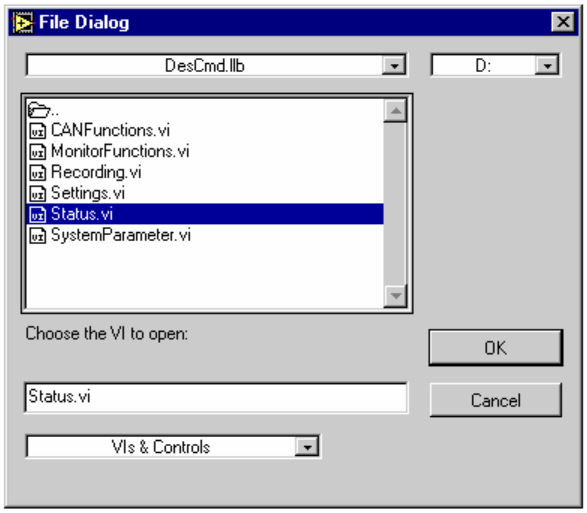


Abbildung 2.3: Öffnen von VI Bibliotheken

Dritter Schritt: Öffnen Sie das Diagrammfenster, durch Anwählen des Menupunktes 'Diagramm' im Menü 'Fenster'. Dort befinden sich die bereits vordefinierten DES Befehle als SubVIs. Diese können kopiert und in das eigene Programm eingefügt werden.

Vierter Schritt: Wählen Sie in der Werkzeugpalette das Werkzeug 'Connector' aus und verbinden Sie die verschiedenen Ein- und Augänge des SubVIs direkt mit den eigenen Elementen in Ihrem LabView Programm.

3. Programmierung

Im folgenden Kapitel wird zuerst etwas Grundsätzliches über die typische Abfolge der Funktionsaufrufe erklärt. Im Weiteren werden die Beispielprogramme kurz erläutert, die mit der *DesCmd.dll* (Windows Data Link Library) mitgeliefert werden.

3.1. Grundsätzlicher Programmablauf

Um die Kommunikation mit der DES korrekt zu konfigurieren, müssen eine Reihe von Funktionen vor der ersten Kommunikation mit der DES aufgerufen werden. Der Ablauf sieht folgendermassen aus.

Initialisierungsvorgang
Diese Funktion muss am Anfang des Programmstarts ausgeführt werden.

Funktion	Bemerkung
DES_InitCommunication(...,...,...,...)	Initialisiert die serielle Schnittstelle, mit den vom Benutzer eingegebenen Daten.

Kommunikation mit der DES
Es können beliebige Funktionen ausgeführt werden.

Funktion	Bemerkung
DES_SetVelocity()	Bestimmt den Sollwert der Rotorumdrehungen.
DES_ReadTempParam(...,...,...,...)	Liest die temporären Systemparameter.
etc.	

Abschlussvorgang
Vor dem Beenden des Programms muss die serielle Schnittstelle wieder freigegeben werden.

Funktion	Bemerkung
DES_CloseCommunication(...)	Gibt die serielle Schnittstelle für andere Anwendungen wieder frei.

3.2. Microsoft Visual C++ 6.0 Beispiel

Die Beispielanwendung 'Demo_WinDLL' in Visual C++ ist eine dialogbasierte Anwendung. Sie zeigt, wie die Kommunikation mit der DES initialisiert werden muss. Dabei wird die Kommunikation auf COM1 und 38400 Baud voreingestellt. Falls diese Einstellungen anders konfiguriert werden müssen, so muss der Quellcode abgeändert werden und das Beispielprogramm neu kompiliert werden.

Die ganzen Initialisierungen werden in der Memberfunktion 'OnInitDialog()' der Klasse 'Demo_WinDllDlg' vorgenommen. Die serielle Schnittstelle wird am Schluss in der Funktion 'DestroyWindow()' wieder freigegeben.

Mit den Bedienknöpfen können die Funktionen „DES_InitCommunicationDlg", „DES_Enable" und „DES_SetVelocity" ausgeführt werden.

Mittels eines ‚Timers' wird während des Programmablaufes alle 200ms der Status mit der DLL Funktion 'UpdateStatus()' neu gelesen. Ist die Kommunikation gestört, so wird der Timer automatisch gestoppt und es erscheint eine Fehlermeldung.

Falls das Projekt mit einer alten Visual C++ Version verwendet wird, so muss das Projekt neu erstellt werden und kann nicht direkt übernommen werden.

3.3. Visual Basic 6.0 Beispiel

Die Beispielanwendung in Visual Basic 6.0 führt dieselben Funktionen durch, wie die anderen Beispielanwendungen. Beim Starten der Anwendung wird die Kommunikation standardmässig auf COM1 und 38400 Baud konfiguriert.
Während dem Ablauf wird der Status zyklisch gelesen und im Dialog angezeigt. Mit den Bedienknöpfen können die Funktionen „DES_InitCommunicationDlg", „DES_Enable" und „DES_SetVelocity" ausgeführt werden. Ist die Kommunikation gestört, so wird der Timer automatisch gestoppt.

Die Bibliothek DesCmd.dll wird in der Reihenfolge: Arbeitsverzeichnis, Windows-Verzeichnis, Suchpfad gesucht. Sollte sie einmal nicht gefunden werden, muss sie in eines dieser Verzeichnisse kopiert werden.

Falls irgendwelche Probleme mit verschiedenen Versionen auftreten, so soll mit der vorhandenen Version ein neues Projekt generiert werden. Danach kann analog zum Beispielprojekt die Kommunikation programmiert werden.

3.4. Borland C++ Builder 5.0 Beispiel

Das Demoprogramm für Borland C++ Builder hat ebenfalls dieselbe Oberfläche wie die vorangegangenen Beispielprogramme.
Die Standardwerte für die Kommunikationseinstellungen sind auch hier COM1 und 38400 Baud.

Durch betätigen der Bedienknöpfe kann wiederum die DES aktiviert oder deaktiviert werden. Danach kann der Motor gestartet und wieder angehalten werden.

Zu beachten gibt es, dass die Bibliothek (DesCmd.lib) nicht dieselbe ist, wie die von Microsoft Visual C++ kreierte. Das Format ist hier OMF und ist entsprechend für Borland angeglichen.

Falls irgendwelche Versionenkonflikte bestehen, so kann anhand dieses Beispielprogrammes ein neues Projekt generiert werden.

3.5. Borland Delphi 4.0 Beispiel

Die Beispielversion in Borland Delphi hat ebenfalls dasselbe Aussehen wie die vorangegangenen Beispiele. Beim Starten des Programms erscheint ein Dialog, mit den Zuständen der DES. Durch das Betätigen der Bedienknöpfe können ebenfalls Aktionen ausgelöst werden. So kann die DES deaktiviert und aktiviert werden oder der Motor kann mit einer voreingestellten Drehzahl rotieren und wieder gestoppt werden.
Auch hier sind die Standardwerte (COM1, 38400 Baud) für die Kommunikation schon voreingestellt.

Falls irgendwelche Versionenkonflikte bestehen, so kann anhand dieses Beispielprogrammes ein neues Projekt erstellt werden.

3.6. National Instruments LabView 6.0 Beispiel

Das Beispielprogramm in LabView demonstriert, wie die verschiedenen Funktionsblöcke in ein eigenes Programm eingebunden werden können.
Durch die Sequenzstruktur wird sichergestellt, dass der Initialisierungsbefehl („DES_InitCommunication“) am Anfang des Programmstarts aufgerufen wird und am Ende die Abschlussfunktion „DES_CloseCommunication“ durchgeführt wird.
Während des Programmablaufes wird eine Schleife durchlaufen, bis der Anwender den Schalter „On / Off“ betätigt. In dieser Schlaufe kann die DES deaktiviert und aktiviert, sowie der Motor ein- und ausgeschaltet werden.
Zum Demoprogramm in LabView gehören folgende Dateien:

- \LabView6.0\DemoDll.vi
- \LabView6.0\DesCmd.dll
- \LabView6.0\InitCommunication.vi
- \LabView6.0\Enable.vi
- \LabView6.0\SetTempParam.vi
- \LabView6.0\ReadTempParam.vi
- \LabView6.0\SetVelocity.vi
- \LabView6.0\ReadSysStatus.vi
- \LabView6.0\CloseCommunication.vi

Um eine korrekte Kommunikation sicherzustellen, müssen die Werte in den SubVIs entweder nach aussen verknüpft oder intern richtig gesetzt sein.

4. Anhang

4.1. DES System-Parameter

Nr.	Parameter	Länge	Zugriff	Default	Bereich	Einheit
0	Baudrate	16-bit	Lesen/Schreiben	3	9600, 14400, 19200, 38400, 57600, 115200, Bereich: 0 .. 5	
1	System Konfiguration	16-bit	Lesen/Schreiben	1	Siehe Daten-Strukturen	
2	Strom-Regler P-Gain	16-bit	Lesen/Schreiben	3057	0 ... 32767	
3	Strom-Regler I-Gain	16-bit	Lesen/Schreiben	994	0 ... 32767	
4	Max. Ausgang des Strom-Reglers	16-bit	Lesen/Service Schreiben	32512	0 ... 32767	
5	Geschwindigkeits-Regler P-Gain	16-bit	Lesen/Schreiben	682	0 ... 32767	
6	Geschwindigkeits-Regler I-Gain	16-bit	Lesen/Schreiben	220	0 ... 32767	
7	Interner Parameter 1	16-bit	Lesen/Schreiben	2200	ändert nichts	
8	Interner Parameter 2	16-bit	Lesen/Schreiben	729	ändert nichts	
9	Interner Parameter 3	16-bit	Lesen/Schreiben	13640	ändert nichts	
10	Beschränkung des Geschw.-Fehlers für den Eingang des Geschw.-Reglers	16-bit	Lesen/Service Schreiben	32767	0 ... 32767	
11	Verstärkung der gesetzten Einheit	16-bit	Lesen	24576	0 ... 32767	
12	Abgleich der gesetzten Einheit	16-bit	Lesen/Schreiben	0	-100 ... +100	
13	Abweichung der gesetzten Einheit	16-bit	Lesen/Service Schreiben	0	0 ... 32767	
14	Spitzenstrom	16-bit	Lesen/Schreiben	15000	1 ... 15000	mA
15	Max. Dauerstrom	16-bit	Lesen/Schreiben	5000	1 ... 5000	mA
16	Thermische Konstante	16-bit	Lesen/Service Schreiben	30400	0 ... 32767	
17	Max. Geschwindigkeit	16-bit	Lesen/Schreiben	25000	0 ... 25000	rpm
18	Beschleunigung	16-bit	Lesen/Schreiben	32000	0 ... 32767	(rpm/128ms)
19	Geschwindigkeits-konstante	16-bit	Lesen/Service Schreiben	0	0 ... 32767	rpm/V
20	Encoderauflösung	16-bit	Lesen/Schreiben	500	0 ... 32767	pulse/turn
21	Anzahl Polpaare	16-bit	Lesen/Schreiben	1	1 ... 64 Standard 1 Polpaar Flachmotor x Polpaare	

Nr.	Parameter	Länge	Zugriff	Default	Bereich	Einheit
22	Interner Parameter 4 (ändert nichts)	16-bit	Lesen/Service Schreiben	960	ändert nichts	(rpm*ms)/ (32*qc)
23	Umrechungsfaktor: rpm in qc/ms	16-bit	Lesen/Service Schreiben	2189	0 ... 32767	qc/(65535*rpm* ms)
24	Winkel des indizierten Pulses	16-bit	Lesen/Service Schreiben	0	-32768 ... 32767	qc
25	PWM Takt	16-bit	Lesen	394	-32768 ... 32767	clock
26	Max. Arbeitszyklus	16-bit	Lesen/Service Schreiben	7373	0 ... 32767	
27	Phasenabgleich u	16-bit	Lesen/Service Schreiben	32512	0 ... 65535	
28	Phasenabgleich v	16-bit	Lesen/Service Schreiben	32512	0 ... 65535	
29	Abgleich AD-Wandler	16-bit	Lesen/Service Schreiben	32768	0 ... 65535	
30	CAN Module ID	16-bit	Lesen	2	1 ... 2047	
31	CAN Service ID	16-bit	Lesen	127	1 ... 2047	
32	CAN RPDO ID	16-bit	Lesen/Schreiben	513	1 ... 2047	
33	CAN TPDO ID	16-bit	Lesen	385	0 ... 2047	
34	CAN BCR1	16-bit	Lesen	1315	0 ... 65535	
35	CAN BCR2	16-bit	Lesen	1	0 ... 65535	
36	CAN Operation Modus (nicht benötigt)	16-bit	Lesen/Service Schreiben	0	0 ... 65535	
37	CAN RxSDO ID	16-bit	Lesen	1537	1537...1663 ID = 1536 + moduleID	
38	CAN TxSDO ID	16-bit	Lesen	1409	1409...1535 ID = 1408 + moduleID	
39	CAN RTR0 ID	16-bit	Lesen	386	385...1407	
40	CAN RTR1 ID	16-bit	Lesen	387	385...1407	
41	CAN Konfiguration	16-bit	Lesen	0	Siehe Daten-Strukturen	
42	InternalParam5	16-bit	Lesen	0	ändert nichts	
43	ErrorProc	16-bit	Lesen/Schreiben	0	0: Disable 1: Stopp 0 ... 1	
44	MaxSpeed in Current Regulation Mode	16-bit	Lesen/Schreiben	30000	0 ... 32767	rpm
45	HallAngleOffs	16-bit	Lesen/Service Schreiben	0	-32768 ... 32767	qc

Notizen:

Lesen = der Wert des Parameters kann gelesen werden
Schreiben = der Benutzer hat Zugriff, um den Parameter zu überschreiben
Service Schreiben = der Benutzer hat nur Zugriff, um den Parameter zu überschreiben, wenn die DES in den Service Modus gesetzt wurde (siehe Kommando *Service*)

4.2. DES Zustands-Variablen

Nr.	Variable	Länge	Einheit
128	Systemzustand	16-bit	Siehe Kapitel 'Datenstrukturen'
129	Ermittelter Stromeffektivwert des Real-Anteils	16-bit	mA
130	Ermittelter Stromeffektivwert des Imaginäranteils	16-bit	mA
131	Gesetzte Stromamplitude	16-bit	mA
132	Relative Rotorposition einer Umdrehung	16-bit	qc
133	Gesetzte Drehzahl	16-bit	rpm
134	Aktueller Mittelwert der Drehzahl	16-bit	rpm
135	reserviert		
136	Absolute Rotorposition	32-bit	qc
137	Standard Fehler	16-bit	Siehe Kapitel 'Standard Error Message'
138	CAN Fehler	16-bit	Siehe Kapitel 'CAN Error Message'
139	Aktueller Realwert des Stromes, => Drehmoment. (kein Durchschnittswert)	16-bit	mA
140	Aktuelle Geschwindigkeit (kein Durchschnittswert)	16-bit	rpm
141	Error History 1	16-bit	Siehe Kapitel 'Standard Error Messages'
142	Error History 2	16-bit	Siehe Kapitel 'Standard Error Messages'
143	Encoder Counter	16-bit	qc
144	Encoder Counter at last index	16-bit	qc
145	Hall sensor pattern	16-bit	Siehe Kapitel 'Datenstrukturen'

4.3. Variablen Typen

Name	Datentyp	Bits	Byte	Bereich
char	vorzeichenbehafteter Integer	8	1	-128 ... 127
BYTE	vorzeichenloser Integer	8	1	0 ... 256
short	vorzeichenbehafteter Integer	16	2	-32'768 ... 32'767
WORD	vorzeichenloser Integer	16	2	0 ... 65'535
long	vorzeichenbehafteter Integer	32	4	-2'147'483'648 ... 2'147'483'647
DWORD	vorzeichenloser Integer	32	4	0 ... 4'294'967'295

4.4. Datenstrukturen

Im Folgenden werden die wichtigsten Datenstrukturen, die von den verschiedenen Befehlen verwendet werden definiert. Da die DLLs in der Programmiersprache C++ geschrieben wurden, sind die folgenden Definitionen ebenfalls in der C++-Syntax aufgeführt.

T_ErrHandler

- typedef void (*T_ErrHandler)(BYTE, BYTE, __int16, BOOL);
- BOOL InitDesCommander(T_ErrHandler yourHandler);

4.4.1. Definition des DES_SysParam

```
typedef struct DES_SysParam
{
    short Baudrate;             //ParNb 0:   R/W;  0=9600; 1=14400; 2=19200; 3=38400;
                                                   4=57600; 5=115200 Baud
    short SysConfig;            //ParNb 1;   R/W;  System Konfiguration (siehe Bit Definition)
    short CurRegGainP;          //ParNb 2;   R/W;  Stromregler P-Gain
    short CurRegGainI;          //ParNb 3;   R/W;  Stromregler I-Gain
    short MaxCurOutput;         //ParNb 4;   R/W;  Maximaler Ausgang des Stromreglers
    short SpeedRegGainP;        //ParNb 5;   R/W;  Geschwindigkeitsregler P-Gain
    short SpeedRegGainI;        //ParNb 6;   R/W;  Geschwindigkeitsregler I-Gain
    short InternalParam1;       //ParNb 7;   R/W;  Interner Gebrauch, keine Bedeutung
    short InternalParam2;       //ParNb 8;   R/W;  Interner Gebrauch, keine Bedeutung
    short InternalParam3;       //ParNb 9;   R/W;  Interner Gebrauch, keine Bedeutung
    short MaxSpeedError;        //ParNb 10;  R/W;  Begrenzung des Geschwindigkeitsfehlers
                                                   des Geschwindigkeitsregler-Einganges
    short SettingUnitGain;      //ParNb 11;  R/W;  Verstärkung der gesetzten Einheit
    short SettingUnitOffset;    //ParNb 12;  R/W;  Abgleich der gesetzten Einheit
    short SettingUnitDelay;     //ParNb 13;  R/W;  Abweichung der gesetzten Einheit
    short PeakCurrent;          //ParNb 14;  R/W;  Stromspitze in mA
    short MaxContCurrent;       //ParNb 15;  R/W;  Maximaler Dauerstrom
    short ThermConst;           //ParNb 16;  R/W;  Thermische Konstante
    short MaxSpeed;             //ParNb 17;  R/W;  Maximale Geschwindigkeit
    short Acceleration;         //ParNb 18;  R/W;  Beschleunigung in rpm/ms^2
    short SpeedConstant;        //ParNb 19;  R/W;  Geschwindigkeitskonstante des Motors
    short EncResolution;        //ParNb 20;  R/W;  Encoderauflösung: Abschnitte/Umdrehung
    short PolePairNumber;       //ParNb 21;  R/W;  Anzahl Pol-Paare
    short Qc2RpmFactor;         //ParNb 22;  R/W;  Umrechnungs-Faktor von qc in rpm
    short Rpm2QcFactor;         //ParNb 23;  R/W;  Umrechnungs-Faktor von rpm in qc
    short IndexOffset;          //ParNb 24;  R/W;  Winkel des indizierten Pulses
    short PWM_Period;           //ParNb 25;  R;    PWM periodischer Takt
    short MaxDutyCycle;         //ParNb 26;  R/W;  Maximaler Arbeitszyklus
    short CurDetPhUOffset;      //ParNb 27;  R/W;  Abgleich von (Phase U) Strom-Detektion
    short CurDetPhVOffset;      //ParNb 28;  R/W;  Abgleich von (Phase V) Strom-Detektion
    short ADConvOffset;         //ParNb 29;  R/W;  Abgleich eines gewöhnlichen AD-Wandlers
    short CAN_ModuleID;         //ParNb 30;  R;    CAN Modul-ID
    short CAN_ServiceID;        //ParNb 31;  R;    CAN Service-ID
    short CAN_RPDO_ID;          //ParNb 32;  R/W;  CAN RPDO-ID
    short CAN_TPDO_ID;          //ParNb 33;  R;    CAN TPDO-ID
    short CAN_BCR1;             //ParNb 34;  R;    CAN BCR1
    short CAN_BCR2;             //ParNb 35;  R;    CAN BCR2
    short CAN_OpMode;           //ParNb 36;  R/W;  CAN Operations-Modus
    short CAN_RxSDO_ID;         //ParNb 37;  R;    CAN Empfangs-SDO ID = 1536 + Modul-ID
    short CAN_TxSDO_ID;         //ParNb 38;  R;    CAN Übertragungs-SDO ID
                                                   = 1408 + Modul-ID
    short CAN_RTR0_ID;          //ParNb 39;  R;    Fernübertragungabfrage-ID (Kanal 0)
    short CAN_RTR1_ID;          //ParNb 40;  R;    Fernübertragungabfrage-ID (Kanal 1)
    short CAN_Config;           //ParNb 41;  R;    CAN Konfigurationsregister
    short InternalParam5        //ParNb 42;  R;    Interner Gebrauch, keine Bedeutung
    short ErrorProc             //ParNb 43;  RW;   Fehler Reaktion Verfahren
    short MaxSpeedCurr          //ParNb 44;  RW;   Max. Geschwindigkeit im Strom-Regler
    short HallAngleOffs         //ParNb 45;  R;    Winkel Offset der Hallsensor-Signale
}DES_SysParam;
```


4.4.2. Definition des 'SysConfig' (System Konfiguration)

BIT 0: 0: Geschwindigkeit/Strom bestimmt durch die Software
1: Geschwindigkeit/Strom bestimmt durch den Analog-Eingang ‘Set value’
BIT 1: 0: Beschleunigung aktiviert
1: Beschleunigung deaktiviert
BIT 2: 0: Abhängig von BIT3
1: Stromregulation
BIT 3: 0: Geschwindigkeitsregulation
1: frei
BIT 4: 0: Geschwindigkeits-Anzeige-Signal
1: Drehmoment durch das Anzeige-Signal
BIT 5: ändert nichts
BIT 6: frei
BIT 7: 0: Hält den Motor mit dem digitalen ‘STOP’ an
1: Hält den Motor mit der Software (Kommando ‘StopMotion’) an
BIT 8: 0: Bestimmt die max. Geschwindigkeit durch das Potentiometer ‘P1’
1: Bestimmt die max. Geschwindigkeit durch die Software (Systemparameter Nr. 17)
BIT 9: 0: Bestimmt den Abgleich durch Potentiometer ‘P2’
1: Bestimmt den Abgleich durch die Software (Systemparameter Nr. 12)
BIT 10: 0: Bestimmt den max. Strom durch das Potentiometer ‘P3’
1: Bestimmt den max. Strom durch die Software (Systemparameter Nr. 14 & 15)
BIT 11: 0: Bestimmt die Regelverstärkung (Geschw.-Regler) durch Potentiometer ‘P4’
1: Bestimmt die Regulator-Verstärkung durch die SW (Systemparameter Nr. 5 & 6)
BIT 12: 0: Aktiviert das System durch den digitalen Eingang ‘Enable’
1: Aktiviert das System durch die Software (Kommando ‘Enable’)
BIT 13: 0: Wählt das Monitorsignal durch den digitalen Eingang ‘Digital 1’ aus.
1: Wählt das Monitorsignal durch BIT4
BIT 14: 0: Nicht im Servicemodus (Es ist nicht erlaubt, die adressierten Variablen und Parameter zu überschreiben)
1: Servicemodus (Es ist erlaubt, die adressierten Parameter und Variablen zu überschreiben)
BIT 15: 0: Wählt den Regulationsmodus durch den digitalen Eingang ‘Digital 2’ aus
1: Wählt den Regulationsmodus durch die BIT2, BIT3 and BIT5 aus

4.4.3. Konfiguration des 'Reglermodus'

Stromregler:	SysConfig.Bit2 = 1;	SysConfig.Bit3 = 0;
Geschwindigkeitsregler:	SysConfig.Bit2 = 0;	SysConfig.Bit3 = 0;

4.4.4. Definition des 'Hall sensor pattern'

HallSensorPattern.bit0 : Zustand von Hall Sensor 1
HallSensorPattern.bit1 : Zustand von Hall Sensor 2
HallSensorPattern.bit2 : Zustand von Hall Sensor 3

4.5. Zustands-Anzeige

4.5.1. Definition 'CAN Error Message'

b0:	CAN Error 0	Warn-Zustand
b1:	CAN Error 1	Passiverror-Zustand
b2:	CAN Error 2	Bussaus-Zustand
b3:	CAN Error 3	Bestätigungs-Fehler
b4:	CAN Error 4	Stuff Fehler
b5:	CAN Error 5	Checksummen-Fehler
b6:	CAN Error 6	Zu viele Grund-Fehler
b7:	CAN Error 7	Bitfehler Anzeige
b8:	CAN Error 8	Formale Fehler Anzeige
b9:	CAN Error 9	PDO Zugangsfrequenz ist zu hoch
b10:	CAN Error 10	PDO Kapazitätsüberschreitung
b11:	CAN Error 11	TxPDO keine Bestätigung
b12:	CAN Error 12	TxSDO keine Bestätigung
b13:	CAN Error 13	RxPDO Mitteilung verloren
b14:	CAN Error 14	RxSDO Mitteilung verloren
b15:	CAN Error 15	0 = Übermittlung erfolgreich; 1 = Übermittlungs-Fehler

4.5.2. Definition 'System Operating Status'

BIT 0: 0: Encoder Index nicht gefunden
1: Encoder Index gefunden
BIT 1: 0: Hallsensor-Signal nicht gefunden
1: Hallsensor-Signal gefunden
BIT 2: 0: Rotor-Position nicht gefunden
1: Rotor Position gefunden
BIT 3: 0: System Parameter nicht in EEPROM speichern
1: System Parameter in EEPROM speichern
BIT 4: frei
BIT 5: 0: Vmax/Offset Messung
1: Temperatur-Messung
BIT 6: 0: ±10V SetValue
1: 0 ... 5V SetValue
} Nur Hardware Version 4003h!
BIT 7: 0: Spitzenstrom ist der Maximalstrom
1: Maximalstrom auf max. Dauerstrom begrenzt
BIT 8: 0: im kleinen Strombereich
1: im grossen Strombereich
BIT 9: 0: kein Fehler
1: Fehler vorhanden
BIT 10: 0: Software nicht aktiv
1: Software aktiv
BIT 11: 0: keine Wirkung des Enable Eingangs, der Eingang kann gewechselt werden
1: Enable Eingang aktiviert, der Eingang muss stabil sein
BIT 12: 0: kein Offset im Stromkreis erkannt
1: ein Offset im Stromkreis erkannt
BIT 13: 0: keine Bremse
1: bremsen mit dem maximal gesetzten Strom
BIT 14 +15: 0 + 0: Stromversorgung ist ausgeschaltet
0 + 1: Aktualisiert die Endstufe
1 + 0: Stromversorgung ist eingeschaltet
1 + 1: Stromversorgung ist eingeschaltet

4.5.3. Definition des 'ErrorProc'

Definition der Fehler-Reaktion. Nur die spezifizierten Fehler können konfiguriert werden. Alle anderen Fehler schalten den Antrieb aus.

ErrorProc = 0: Ausschalten der DES bei einem Fehler
ErrorProc = 1: Stoppen der DES bei einem Fehler

Configurable Errors: Error 7: Spannungsversorgung zu tief für den Betrieb
Error 8: Fehler bei der Winkelmessung

4.5.4. Definition des 'CAN Config'

BIT 14: 0: PDO Kanal ausgeschaltet
1: PDO Kanal eingeschaltet
BIT 13: 0: RTR Kanal 1 ausgeschaltet
1: RTR Kanal 1 eingeschaltet
BIT 12: 0: RTR Kanal 0 ausgeschaltet
1: RTR Kanal 0 eingeschaltet

4.5.5. Definition 'Standard Error Message'

b0:	Error 0	Hallsensor-Fehler
b1:	Error 1	Prozessregister-Fehler
b2:	Error 2	Falsche Encoderauflösung
b3:	Error 3	Hallsensor 3 nicht gefunden
b4:	Error 4	Überstrom-Fehler
b5:	Error 5	Überspannungs-Fehler
b6:	Error 6	Drehzahl-Fehler zu hoch
b7:	Error 7	Spannungsversorgung zu tief
b8:	Error 8	Fehler bei der Winkelabfrage
b9:		
b10:		
b11:	Error 11	Übertemperatur (nur Hardware Version 4003h!)
b12:		
b13:	Error 13	Parameter ausserhalb des Bereiches
b14:		
b15:	Error 15	0 = keine Fehler; 1 = Fehler vorhanden

4.6. CAN Bit Timing

Die DES arbeitet bei einer Konfiguration der Bitrate von 1Mbit/s am optimalsten.
Je nach System ändert sich die Bitrate entsprechend (Länge, Teilnehmer). Dies bedeutet auch, dass der Abtastpunkt und die Zeitquanten für den CAN oder CANopen wechseln.
Wenn sie die Bitrate des CAN ändern wollen, so müssen die Register BCR1 und BCR2 neu eingestellt werden. Benützen sie die Funktion ‘SetCAN_BCR1_BCR2' oder die Funktion 'SetCAN_Bitrate' für vorkonfigurierte Werte.

Hier sind die Informationen, damit Sie die zwei Registerwerte berechnen können.

f_{Osc} = 4 * Quarz-Frequenz
TQ = (BRP + 1) / f_{Osc}
$t_{SYNCSEG}$ = SYNCSEG * TQ
T_{TSEG1} = (TSEG1 + 1) * TQ
T_{TSEG2} = (TSEG2 + 1) * TQ
Bit Time = $t_{SYNCSEG}$ + T_{TSEG1} + T_{TSEG2}

SYNCSEG = 1
SJW = 0 - 3
TSEG1 = 2 - 15
TSEG2 = 1 - 7

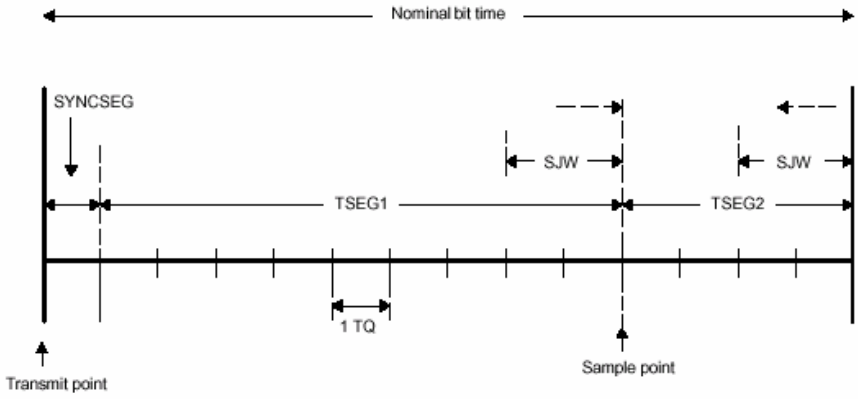


Abbildung 4.1: Bit Timing Berechnung

Definitionen:

$T_{SEG1} \geq T_{SEG2}$
$T_{SEG2min}$ = 1 + SJW

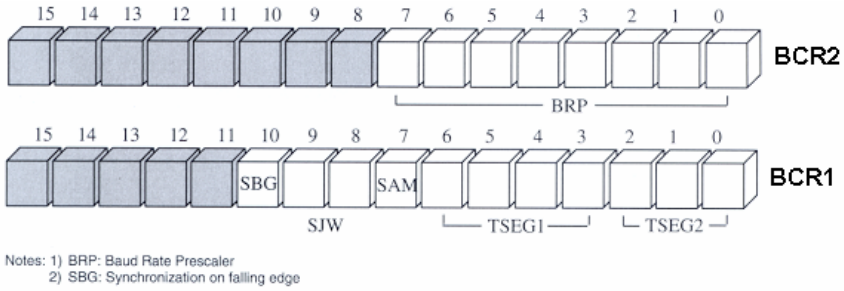


Abbildung 4.2: Bit Timing Register

BRP = Baudrate Prescaler
SBG = 0 (Synchronisation auf fallende Flanke); 1(Synchronisation auf steigende Flanke)
SJW = Synchronisationssprungweite
SAM = 0 (CAN-Modul nur einmal abgetastet)
TSEG1 = Zeitsegment 1
TSEG2 = Zeitsegment 2

Berechnungsbeispiel für 500kBit/s:

Quarz-Frequenz = 5 MHz (Hardware Version 0x4001, 0x4002 and 0x4101)
f_{Osc} = 4 * Quarz-Frequenz = 20MHz
Bitrate = 500 kBit/s
Nominal-Bit-Zeit = 1/Bitrate = 2µs
Anzahl Zeitquanten = 20
Nominal TQ = 100ns

BRP (= BCR2) = (Nominal TQ * f_{Osc}) – 1 = 1
TQ = (BRP + 1) / f_{Osc} = 100ns
$t_{SYNCSEG}$ = 1 * TQ = 100ns
TSEG1 = 15
TSEG2 = 2
T_{TSEG1} = (TSEG1 + 1) * TQ = 1,6µs
T_{TSEG2} = (TSEG2 + 1) * TQ = 300ns
Bit Time = $t_{SYNCSEG}$ + T_{TSEG1} + T_{TSEG2} = 2µs
SJW = 1
SAM = 0
SBG = 0

BCR1 = 017Ah; BCR2 = 0001h

Richtwerte für die Register BCR1 und BCR2

Hier sind einige Bit Timing Werte für Sie vorausberechnet. Die eingestellten Werte dienen einzig als Referenz. Sie müssen die Werte entsprechend Ihrem Netzwerk anpassen.

Tabelle für DSP mit 10MHz Quarz-Frequenz (HW 0x4003):

Bitrate	1MBit/s	800kBit/s	500kBit/s	250kBit/s	125kBit/s	50kBit/s	20kbit/s	10kbit/s
Max Leitungslänge [m]	25	50	100	250	500	1000	2500	5000
BCR1 (hexadezimal)	0h017A	0h0031	0h017A	0h017A	0h017A	0h0173	0h0173	0h0173
BCR2 (hexadezimal)	0h0001	0h0004	0h0003	0h0007	0h000F	0h0027	0h0063	0h00C7

Tabelle für DSP mit 5MHz Quarz-Frequenz (HW 0x4001, 0x4002 and 0x4101):

Bitrate	1MBit/s	800kBit/s	500kBit/s	250kBit/s	125kBit/s	50kBit/s	20kbit/s	10kbit/s
Max Leitungslänge [m]	25	50	100	250	500	1000	2500	5000
BCR1 (hexadezimal)	0h017A	0h017F	0h017A	0h017A	0h017A	0h0173	0h0173	0h0173
BCR2 (hexadezimal)	0h0000	0h0000	0h0001	0h0003	0h0007	0h0013	0h0031	0h0063

Tabelle für DSP mit 4.9152MHz Quarz-Frequenz (HW 0x0001):

Bitrate	1MBit/s	800kBit/s	500kBit/s	250kBit/s	125kBit/s	50kBit/s	20kbit/s	10kbit/s
Max Leitungslänge [m]	25	50	100	250	500	1000	2500	5000
BCR1 (hexadezimal)	0h062A	0h0633	0h063B	0h063B	0h063B	0h0643	0h063C	0h0652
BCR2 (hexadezimal)	0h0001	0h0001	0h0002	0h0005	0h000B	0h001B	0h0045	0h0082
Effektive Bitrate [Bit/s]	983040	819200	504123	252062	126030	50155	20062	10005

