Titel:

Erstellen und Einbinden von externen C-Bibliotheksfunktionen für die 80386 Objektcodegenerierung

Version 0.4

INHALTSVERZEICHNIS

1 ALLGEMEINES 2

2 ABLAUF BEIM ERSTELLEN EINER EXTERNEN C-BIBLIOTHEK 3

3 EXTERNE BIBLIOTHEKSFUNKTIONEN IM LAUFZEITSYSTEM 5

1 Allgemeines

CoDeSys besitzt eine Schnittstelle zum Einbinden von benutzerdefinierten Funktionen und Funktionsbausteinen, die in Ansi-C realisiert sind. Diese sogenannten externen Funktionen können zu einer Bibliothek zusammengefaßt werden und in ein IEC 1131 Programm eingebunden werden.

Diese Schnittstelle erlaubt somit bestehende C-Bibliotheken (z.B. Regelungsbausteine, Kommunikationstreiber,...) weiterzuverwenden.

Bei externen Bibliotheken muß unterschieden werden, ob sich die Funktionen direkt in der externen Bibliothek oder im Laufzeitsystem befinden.

Das folgende Kapitel beschreibt die Vorgehensweise beim Erstellen einer externen C-Bibliothek, deren Funktionen in der Bibliothek ausprogrammiert sind.

Kapitel 3 beschreibt das Erstellen einer externen Bibliothek, deren Funktionen sich im Laufzeitsystem befinden. Hierbei beschreiben die Bibliotheksfunktionen nur die Schnittstelle.

2 Ablauf beim Erstellen einer externen C-Bibliothek

1. Erzeugen Sie mit CoDeSys ein neues Projekt, das die Bibliotheksfunktionen enthält (Funktionen und Funktionsbausteine). Definieren Sie die Schnittstelle und den Code, der im Simulationsmodus ausgeführt werden soll.
2. Speichern Sie dieses Projekt als Bibliothek mit dem Menü <Datei><Speichern unter...> und Auswahl des Dateityps <Externe Bibliothek> in der Dialog-Box. Speichern Sie es auch als normales Projekt, um die Bibliothek auch in Zukunft weiter pflegen oder weitere Funktionen einbinden zu können.
   Beim Speichern als externe Bibliothek wird eine Datei <Libname>.h und <Libname>.lib erzeugt.
3. Erzeugen Sie eine C-Datei aus der Header-Datei <Libname>.h.
   Bei Funktionsbausteinen wird zusätzlich zu der eigentlichen Bibliotheksfunktion noch eine zugehörige Initialisierungsfunktion generiert. Darin müssen Sie die Daten des Funktionsbausteins initialisieren.
   Beachten Sie, daß die Reihenfolge der Funktionen in der C-Datei gleich sein muß wie im Projekt (alphabetisch geordnet). Die Reihenfolge der Funktionen in der Header-Datei muß bestehen bleiben!
   Die Typdefinitionen wie z.B: BOOL, die in der Header-Datei verwendet werden, befinden sich in der Datei lzstyp.h. Beachten Sie, daß diese in Konflikt mit z.B. windows.h stehen können!! Sie sollten nicht beide Header-Dateien includieren. Falls dies unvermeidbar ist: Ein BOOL-Parameter in einer CoDeSys-Bibliotheksfunktion muß ein char sein. Notfalls schreiben Sie explizit char!
   Beachten Sie, dass alle referenzierten Funktionen im C-File enthalten sein müssen.
   Die Initialisierung von global deklarierten C-Variablen mit Konstanten ist möglich:
   int i = 5;
   Die Initialisierung mit Ausdrücken aber nicht:
   int pi = &i; /* nicht möglich */
   Initialisierungen, die einen Ausdruck benötigen, müssen darum in einer Init-Funktion durchgeführt werden.
4. Programmieren Sie die Bibliotheksfunktionen aus und legen Sie mögliche Hilfsfunktionen ans Ende der Datei.
5. Erstellen Sie mit Visual C++ (oder Ihrem C-Compiler) ein Projekt für eine Windows- oder Consolen-Applikation mit dem Namen der Bibliothek (<Libname>.mdp) und fügen Sie die Bibliotheks C-Datei in das Projekt ein.

   Achtung: Folgende Compilereinstellungen sind in Visual C++ notwendig:

   - Project / Settings / C/C++ / Customize: Enable function-level linking deaktivieren
   - Project / Settings / C/C++ / Optimization: Customize: Global Optimizations + Generate Intrinsic Functions + Favor small code aktivieren
   - Project / Settings / C/C++ / Code Generation: Struct member alignment = 1 Byte
   - Wenn Sie irgendeine Ansi-C Standardbibliotheksfunktion verwenden wollen, müssen Sie in den Projekteinstellungen folgende Auswahl aktivieren:
     Build / Settings / C/C++ / Category: Optimizations : Customize: Generate Instrinsic Functions
     Damit werden die einfachsten Ansi-C Bibliotheksfunktonen Inline aufgelöst (strcpy(),...).Ansonsten dürfen keine Bibliotheken verwendet werden.
   - Für Visual 6.0:
     Project / Settings / C/C++ / General: Debug Info: Program Database (nicht: Program Database for Edit and Continue)
6. Kompilieren Sie danach das Projekt. Wenn die Bibliothek korrekt ist, erscheint nur die Fehlermeldung: „Unresolved external symbol WinMain“ bzw. „main““.
   Danach existiert ein Object-File zum Einbinden in Ihr IEC 1131 Projekt (<Libname>.obj).
7. Kopieren Sie das erzeugte Object-File (<Libname>.obj) und die Bibliothek (<Libname>.lib) in das Bibliotheksverzeichnis von CoDeSys.

8. Binden Sie die Bibliothek in Ihr IEC 1131 Programm mit der Bibliotheksverwaltung von CoDeSys ein (Fenster/Bibliotheksverwaltung, Einfügen/Bibliothek).
9. Übersetzen Sie Ihr Projekt mit der externen Bibliothek um zu überprüfen, ob Sie die oben beschriebenen Vorgänge korrekt ausgeführt haben.
10. Nun können Sie in Ihrem Projekt auf alle Funktionen Ihrer Bibliothek zugreifen.

3 Externe Bibliotheksfunktionen im Laufzeitsystem

In diesem Kapitel wird beschrieben, wie eine externe Bibliothek erstellt wird, deren Funktionen sich im Laufzeitsystem befinden. Hierbei beschreiben die Funktionen in der Bibliothek nur die Schnittstellen der Funktionen (Prototypen).

1. Erzeugen Sie mit CoDeSys ein neues Projekt, das die Bibliotheksfunktionen enthält (Funktionen und Funktionsbausteine). Definieren Sie die Schnittstelle und programmieren Sie die Funktionen aus, die nur in der Simulation verwendet werden (siehe Kapitel 2., Punkt 1.).
2. Speichern Sie dieses Projekt als Bibliothek mit dem Menü <Datei><Speichern als Bibliothek...> und Auswahl des Buttons <Externe Bibliothek> in der Dialog-Box. Speichern Sie es auch als normales Projekt, um die Bibliothek auch in Zukunft weiter pflegen oder weitere Funktionen einbinden zu können.
   Beim Speichern als externe Bibliothek wird eine Datei <Libname>.h und <Libname>.lib erzeugt. Die Header-Datei beinhaltet die Schnittstellen-Definition zu den Funktionen im Laufzeitsystem.
3. Die Integration der Bibliotheksfunktion in die SoftPLC erfolgt im Modul RtsCst.c.

Die Vorgehensweise soll anhand einer Bibliotheksfunktion (CstDummy) veranschaulicht werden:

a) Implementierung der Bibliotheksfunktion im Modul RtsCst.c

b) Erweitern der Referenz-Tabelle (CstExtRefTable) um den Eintrag der neuen Bibliotheksfunktion. Die erste Struktur-Komponente muß den Namen der Bibliotheksfunktion beinhalten, wie Sie in der IEC-Bibliothek (siehe oben) eingegeben wurde. Dieser Name ist auf maximal 31 Zeichen begrenzt.
Der zweite Eintrag muß den Funktionszeiger auf die Funktion liefern.

c) mit der Funktion CstGetExtRefTable muß nun ein Zeiger auf die Refernztabelle zurückgegeben werden

Nachfolgend ist der zugehörige Codeausschnitt abgebildet:

```
static void CstDummy(void)
{
    /* TODO: Implement your library-function */
    return;
}

static ExtRef CstExtRefTable[] =
{
    /*{Function-name (max. 31 chars), Function-pointer}*/
    {"Dummy", (void (*)())CstDummy},
    {"", NULL}
};

ExtRef* CstGetExtRefTable(void)
{
    return CstExtRefTable;
}
```