**Software-Funktionsbeschreibung**

**STM IGEMS-Interface**

**Anlage** :

**Lieferant** : NUM AG

Battenhusstrasse 16

Postfach Lustmühle

CH - 9053 Teufen

Telefon: 071 / 335 04 11

Telefax: 071 / 333 35 87

**Dokument** : Zuständig V. Fleischer

Dateiname FB-IGEMS-Interface.docx

Datum 14.05.2018

SW-Version V1\_0

# Konzept

STM möchte gern, dass wir als NUM eine Schnittstelle zur Steuerung anbieten die unabhängig von den verschiedenen Flexium-Versionen ist, damit IGEMS nicht nach einem Update wieder angepasst werden muss.

IGEMS hat vorgeschlagen die Schnittstelle in Form einer C# Assembly auszuführen. Daraufhin wurde von unserer Entwicklung ein Prototyp einer solchen Proxy-DLL erstellt, die auf den von STM gewünschten Funktionsumfang erweitert wurde.

Der gewünschte Funktionsumfang wurde in Zusammenarbeit mit STM und NUM AT festgelegt. Die bisherige Funktionsweise eines endlos laufenden Hauptprogramms, dass nacheinander einzelne Job aufruft, soll weiter beibehalten werden. Dies ist notwendig, da in den einzelnen Job mitunter Aufgaben wie „Schneidkopf absenken“, „Wasser & Abrasivmittel ein/aus“ etc. aufgerufen werden können. Nach Abschluss des Jobs sollen die angewählten Funktionen dann weiter aktiv bleiben. Nach einen Programm-Reset wäre das aber nicht der Fall.

Damit die Funktionen NC-Start, NC-Stop und NC-Reset, Modusanwahl nicht direkt ohne Freigabe von der SPS aufgerufen werden können, sollen diese Funktionen durch symbolische Variablen in die SPS übertragen und vor dort aus aktiviert bzw. unterdrückt werden.

Die von STM gewünschten Funktionen sind:

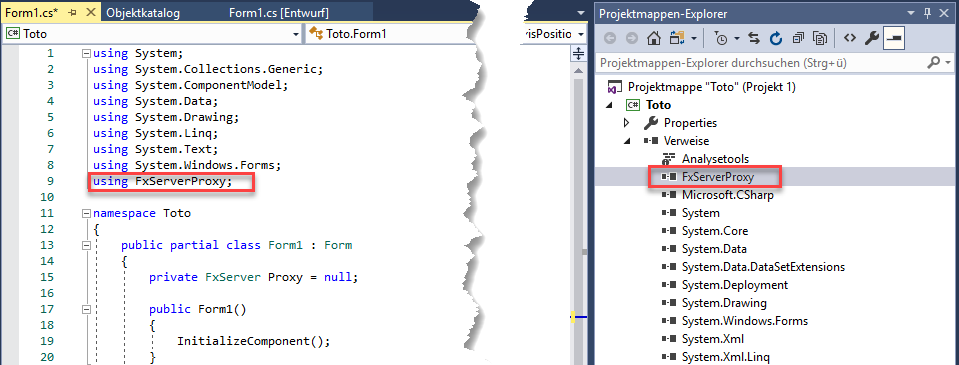
* Modusanwahl: HAND / AUTO
* Programm laden und aktivieren (für Hauptprogramm oder „normale“ Programme)
* Programm Start, Stopp, Reset
* Job nachladen und aktivieren
* Istposition der Maschine abfragen

Das „Nachladen“ von Jobs wurde bisher mit einer FxServer-Anwendung „NumBlockload“ durchgeführt. Diese Anwendung nimmt neue Programme von IGEMS entgegen und lädt sie in die Steuerung. Es gibt verschiedene Versionen des Nachladens. In einer Version wurden die Jobs auf verschiedene Unterprogrammnummern verteilt, je nachdem ob bereits aktuell ein Job ausgeführt wurde. Da es aber bei der Verwendung von Flexium 3D Probleme gibt, wenn die Schneidprogramme unter abwechselnden Unterprogrammnummern ausgeführt werden, wurde eine weitere Version entwickelt, die erst Unterprogramme nachlädt wenn ein bereits laufender Job beendet ist. Nachdem das Unterprogramm geladen wurde, wird dieser Job wird über das Beschreiben einer E8xxxx-Variablen mit der zu startenden Unterprogramm-Nummer aktiviert.

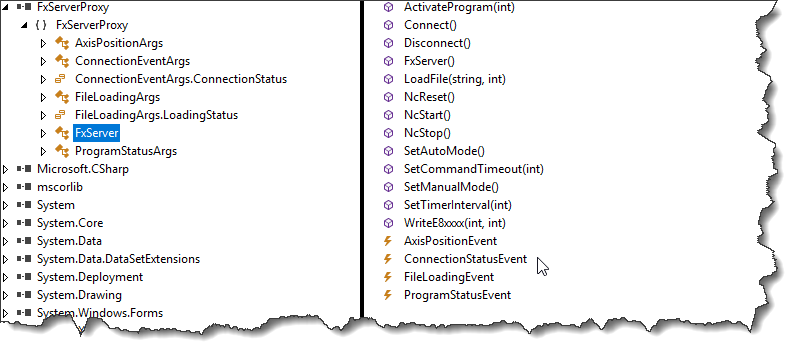
# Funktionen Beschreibung

## Proxy-DLL in C#-Anwendung einbinden

Damit man mit der DLL eine Verbindung zur Steuerung aufbauen kann, muss ein Verweis in die entsprechende Anwendung aufgenommen werden.

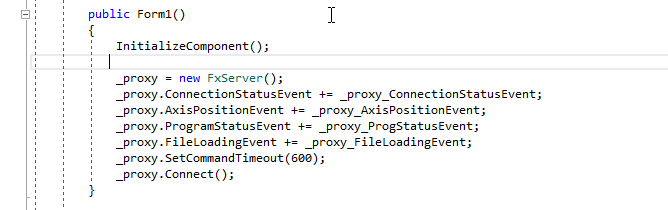


Die FxServer Klasse der Proxy-DLL stellt momentan die folgenden Methoden und Events bereit:



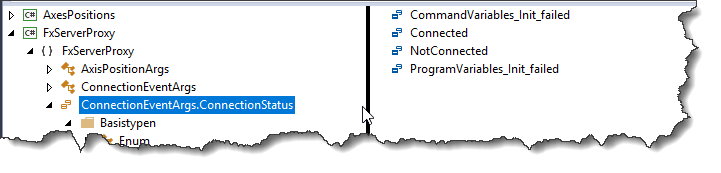
## Verbindung zur Steuerung aufbauen und trennen

Um eine Verbindung mit der Steuerung aufnehmen zu können, muss die Klasse instanziiert und die gewünschten Events verlinkt werden. Danach muss die Connect-Methode aufgerufen werden.



Optional kann die Zeit bis zur Rücksetzung der Kommandovariablen modifiziert werden, wenn der Vorgabewert von 300 ms nicht gewünscht ist.

Nachdem die Verbindung zur Steuerung aufgebaut wurde, wird im ConnectionStatusEvent der Zustand „Connected“ übergeben.



Wenn es Probleme beim Verbinden der Kommandovariablen oder der Programmstatus-Variablen gibt, dann wird ein entsprechendes Ereignis ausgelöst.

Spätestens beim Beenden der Anwendung sollte die Verbindung zur Steuerung wieder getrennt

werden. Dazu die Methode „Disconnect“ aufrufen.

## Aktionen auslösen

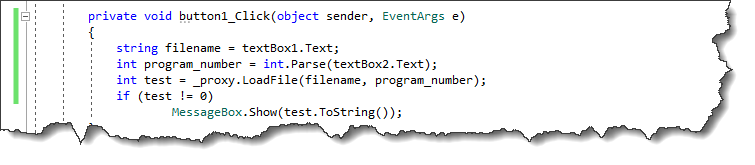
Die Betriebsartenumschaltung sowie die NC-Befehle: “Start“, „Stop“ und „Reset“ werden über symbolische SPS Variablen (siehe Abschnitt 3 PLC) übertragen. Die Proxy-Klasse kapselt diese Funktionalität und stellt dedizierte Methoden zum Starten, Stoppen etc. zur Verfügung. Innerhalb der Methode wird die Statusvariable für eine definierte Zeit „CommandTimeout“ auf 1 gesetzt und anschließend wieder auf 0 zurückgesetzt. Die SPS muss innerhalb dieser Zeit die Anforderung erkennen und entsprechende Aktionen auslösen. Ein Handshaking zur zweifelsfreien Erkennung einer Aktion wurde nicht gewünscht.

Alle Methoden haben einen Rückgabewert. Der Rückgabewert 0 bedeutet, dass die Methode fehlerfrei ausgeführt wurde. Werte kleiner 0 weisen darauf hin, dass die Ausführung wegen nicht erfüllter Bedingung „FxServer nicht verbunden“ (-1) nicht erfolgte. Rückgabewerte größer 0 entsprechen dem vom FxServer gelieferten Fehlerrückgabewert der aufgerufenen Funktion WriteVariables2.

Die Rückgabewerte der Funktion VariablesWritten2 werden der Einfachheit halber nicht ausgewertet, da ein fehlerhaftes Nichtauslösen der Schreibfunktion keine gravierenden Fehler auslösen kann.

## Programm laden und aktivieren

Um ein Programm in die Stuerung zu laden wird die Methode LoadFile zur Verfügung gestellt. Es müssen der Dateiname mit Pfadangabe der zu ladenden Datei angegeben werden und die Programmnummer, unter der diese Datei in der Steuerung abgelegt wird.



Auch hier gibt der Rückgabewert der Methode an, ob das Laden der Datei aktiviert wurde.

Rückgabewerte der Methode sind:

-3: Programmladen wurde nicht gestartet, weil das aktuell ausgeführte Programm überschrieben werden sollte

-2: Es ist noch ein anderer Dateitransfer aktiv

-1: FxServer nicht verbunden oder Filetransferobjekt nicht initialisiert.

0: Programmupload gestartet

>0: Fehlerrückgabewerte des FxServers

Nachdem ein Programm geladen wurde muss es noch aktiviert werden, wenn die aktuelle Programmnummer nicht schon der eben geladenen Datei entspricht. Dazu muss die Methode ActivateProgram verwendet werden. Auch diese Methode gibt zur Kontrolle Rückgabewerte zurück.

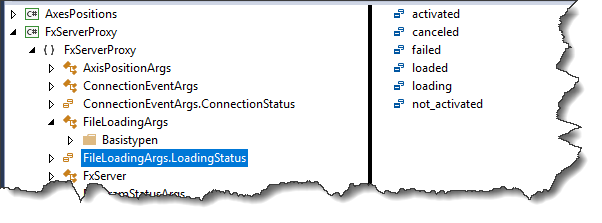
-1: FxServer nicht verbunden oder Generalfunctions-Object nicht initialisiert.

0: Programmaktivierung gestartet

>0: Fehlerrückgabewerte des FxServers

Da das Laden der Datei oder die Aktivierung asynchron erfolgt, wird das erfolgreiche Laden oder der Abbruch des Ladens bzw. der Aktivierung durch einen FileLoadingEvent angezeigt.

Der Ladestatus einer Datei wird durch die folgenden Zustände angezeigt.



## Job aktivieren

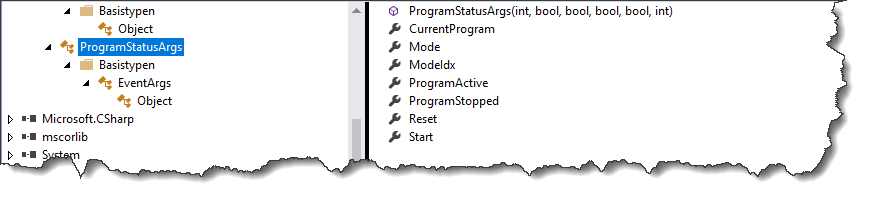
Im speziellen Fall dass das Hauptprogramm mit einer Endlosschleife in der NC ausgeführt wird, welches durch die Prüfung einer E8xxxx Variablen ein Unterprogramm mit der auf diese Variablen geschriebenen Unterprogrammnummer aufruft, kann durch das Beschreiben eben dieser E8xxxx Variablen ein Job aktiviert werden. Dazu wird die Methode WriteE8xxxx(int E8xxxx, int Wert) bereitgestellt. Beim Aufruf muss die E8xxxx Nummer die beschrieben und der Wert der geschrieben werden soll, angegeben werden. Ist die E8xxxx Zahl kleiner als 80000, so wird sie intern um 80000 erhöht.

Das erfolgreiche Aktivieren eines Jobs wird nur implizit durch die Programm-Status Änderung der aktiven Programmnummer im „ProgramStatusEvent“ mitgeteilt.

## Programm Status Event

Wie bereits erwähnt wird über den „ProgramStatusEvent“ der Zustand der NCK über das Interface geliefert.

Der Event liefert ein Objekt mit den folgenden Eigenschaften zurück.



* CurrentProgram zeigt das aktuell ausgeführt Programm (Job) an
* Mode zeigt den NC Modus (Hand, Auto, etc an)
* ModeIdx zeigt den entsprechenden Mode-Index an
* Start zeigt an, dass der NC-Start ansteht
* ProgramActive / ProgramStopped zeigt an ob ein Programm in Ausführung oder im M0 steht
* Reset zeigt an, dass gerade ein Reset in der NC aktiv ist.

## AxisPosition Event

Die Übermittlung der aktuellen Maschinenposition erfolgt über den AxisPositionEvent.

Das im Event übermittelte Objekt vom Typ AxisPositionArgs hat die Eigenschaften:

Idx: Achsindex (0..X, 1..Y, 2..Z, etc. gemäß FxServer SDK)

AxisName: Achsbuchstabe (siehe Idx)

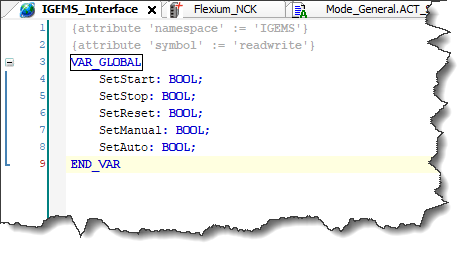
AxisPosition: Aktuelle Position in mm bzw. Grad

Der Einfachheit halber wird immer nur 1 Achse je Event übertragen. Ändern sich mehrere Achsen gleichzeitig, so folgen entsprechend viele Events kurz hintereinander.

# PLC

Wie bereits erwähnt werden zur Kommunikation mit der SPS symbolische Variablen eingesetzt.

Diese symbolischen Variablen müssen wie folgt definiert und mit im „namespace“ IGEMS veröffentlicht werden.



Weiterhin obliegt es der SPS die Signale entsprechend auszuwerten, damit die Aktionen wie Programm-Start / Stop etc. ausgeführt werden. Auf das Handling wird hier nicht näher eingegangen.

# CNC Programme

Anbei das bisher verwendet Hauptprogramm, welches in Verbindung mit NumBlockLoad ein Unterprogramm aufruft, dass durch die im E80004 übergebene Nummer gestartet wird.

%200

E10028=1

IF E10029=0 THEN

G77 H9990

ENDI

N1

E80003=E80004

E80004=0

G79 E80003<>0 N3

G997

G4 F0.1

G79 N1

N3

G4 F1

L199=E80003

G4 F1

G77 HL199

L199=0

G77 H9057

G79 N1

N9

M2

# Modifikationen

|  |  |  |  |
| --- | --- | --- | --- |
| **SW-Version:** | **Datum:** | **Autor** | **Kommentar:** |
| **V1\_0** | 03.05.201 | V. Fleischer |  |

**Änderungen:**