miCon-L

Erste Schritte

ProSign GmbH

Werner-Heisenberg-Straße 1
D-39106 Magdeburg

Telefon: +49 (0) 391 563 068 90
Telefax: +49 (0) 391 563 068 99
E-Mail: support@micon-l.de
Web: www.pro-sign.de

BARTH® Elektronik GmbH

Gewerbepark BW-Depot
Im Depot 1-3
D-49838 Lengerich/Emsland

Telefon: +49 (0) 5904 499 970

E-Mail: info@barth-elektronik.de
Web: www.barth-elektronik.com

Inhalt

1 Einleitung und Ziel des Dokuments

Dieses Dokument stellt wesentliche Aspekte des grafischen Programmiersystems miCon-L 7.1 vor, um damit einen schnellen Einstieg zu ermöglichen. Es richtet sich an Umsteiger von einer älteren miCon-L-Version (<= V3.8.1) aber auch an Neueinsteiger. Gleichzeitig ist es als zentrales Informationsdokument gedacht, um schnell zu weiterführenden Informationen zu kommen.

Das Dokument „Erste Schritte“ ist so aufgebaut, dass es in Kapitel 2 mit grundsätzlichen Informationen beginnt, die im Wesentlichen an Neueinsteiger gerichtet sind.

➔ Kapitel 2 - Neueinsteiger

In Kapitel 3 sind Informationen zu Änderungen und Neuerungen der Version 7.1 und Hinweise zum Projektimport aus älteren Versionen zusammengefasst.

➔ Kapitel 3 - Umsteiger

Ab Kapitel 4 folgen weitere nützliche Informationen als Überblick.

➔ Kapitel 4 - Mehr Informationen

2 Neueinsteiger

2.1 Was ist miCon-L?

miCon-L ist eine graphische Programmiersoftware, die auf dem Programmiersystem iCon-L basiert und speziell für die Programmierung von BARTH lococube® mini-SPS der STG-Serie angepasst wurde.

Es werden Blockdiagramme für die Beschreibung bzw. Modellierung von technischen Systemen genutzt, sodass ein Erlernen textueller Programmiersprachen durch den Anwender nicht erforderlich ist.

Ein Programm, bzw. Anwendungsdesign, wird durch die grafische Verbindung von Funktionsbausteinen erstellt. Die Bausteinreihenfolge legt die Abarbeitungsreihenfolge fest und die Verbindungen visualisieren den Datenfluss. Die Funktionsbausteine werden durch technische Symbole dargestellt, die sehr einfach zu verstehen sind.

Viele Funktionsbausteine sind parametrierbar und z.B. in ihrem Datentyp anpassbar. Solche einfachen Anpassungen werden implizit über die verschiedenen Farben visualisiert. Die Parametrierbarkeit kann aber auch sehr weit reichen und ist stets für den jeweiligen Anwendungszweck zugeschnitten.

Die BARTH lococube® mini-SPS ist eine Gruppe von speicherprogrammierbaren Steuerungen, die in der Industrie, wie beispielsweise in den Bereichen Maschinen- und Fahrzeugbau, breite Anwendung findet. Sie verfügt über kompakte Abmessungen und ist aufgrund ihrer robusten Bauweise auch für den Einsatz in rauen Umgebungen gut geeignet. Bei den aktuellen Modellen besteht die Möglichkeit, Daten über CAN-Bus oder eine Modbus-Slave-Schnittstelle zu übertragen.

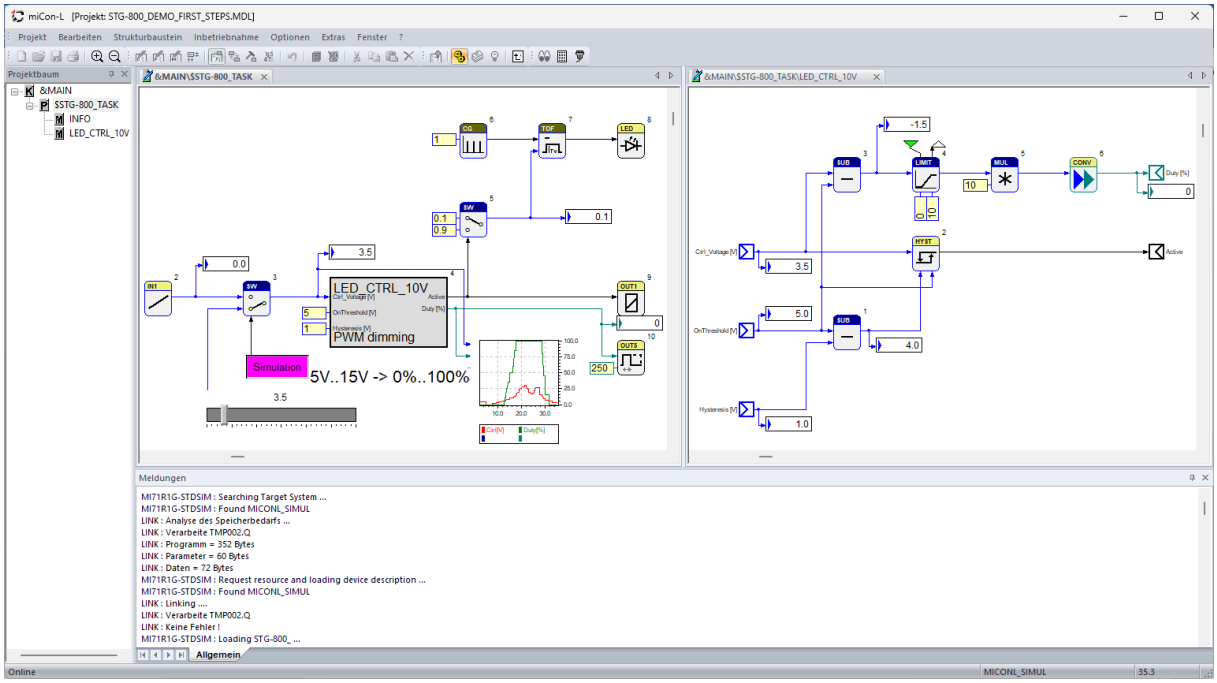


Abbildung 1: miCon-L im Onlinemodus (Beispielprojekt zur LED-Steuerung)

2.2 Benutzeroberfläche von miCon-L

Die Benutzeroberfläche von miCon-L hat die drei grundsätzlichen Modi „Editiermodus“, „Inbetriebnahmemodus“ und „Onlinemodus“.

Befehle können über das Menü, die Symbolleiste, das Kontextmenü (Rechtsklick) und Tastenkombinationen aufgerufen werden.

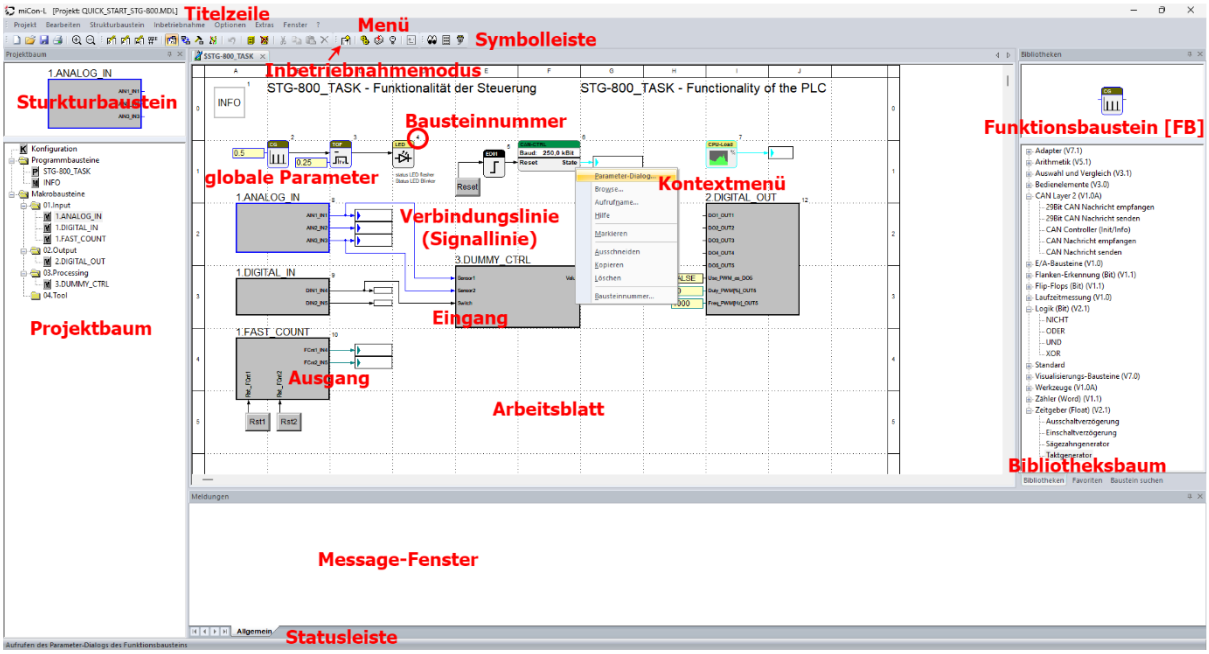


Abbildung 2: Benutzeroberfläche im Editiermodus

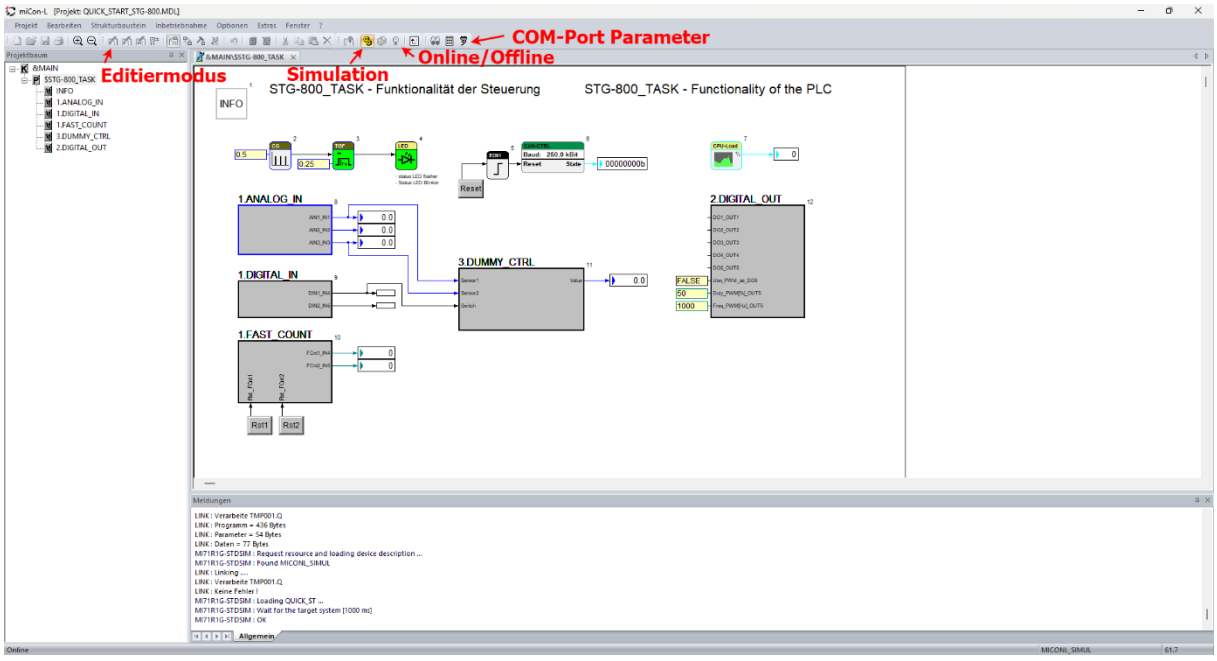


Abbildung 3: Benutzeroberfläche im Onlinemodus

Im Inbetriebnahmemodus und im Onlinemodus können keine Funktionsbausteine oder Strukturbausteine eingefügt werden. Das erstellte Programm kann jedoch parametriert und/oder in ein Zielsystem (z.B. „Simulation“) geladen und beobachtet werden.

2.3 Softwaredownload und Installation

Der Softwaredownload der aktuellen miCon-L-Version wird für die englische und die deutsche Version jeweils über eine eigene Unterseite der miCon-L-Webseite angeboten.

Deutsch: https://barth-elektronik.com/miCon-L/

Englisch: https://barth-elektronik.com/en/miCon-L/

Nach dem Download der aktuellen Version kann diese installiert werden, indem die entsprechende Setup-Datei ausgeführt wird (z.B. miConL71R1DE_Setup.exe).

Abbildung 4: Startseite der Installation

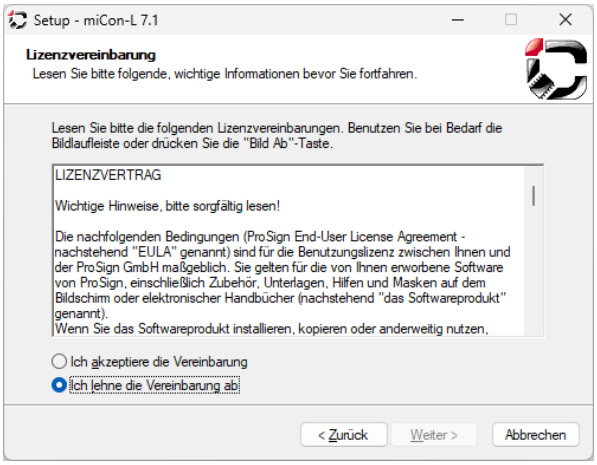


Abbildung 5: Lizenzvereinbarung

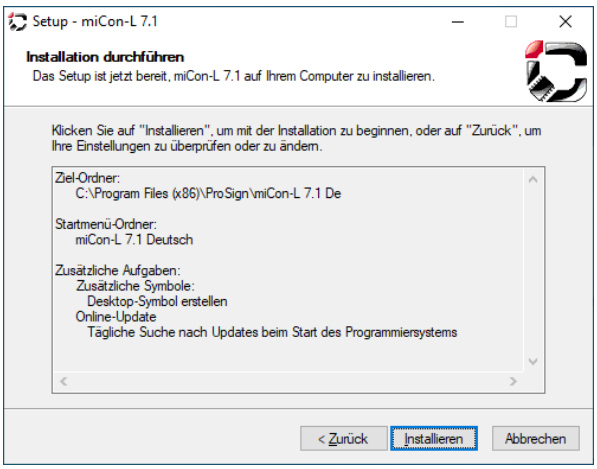


Abbildung 6: Zusammenfassung

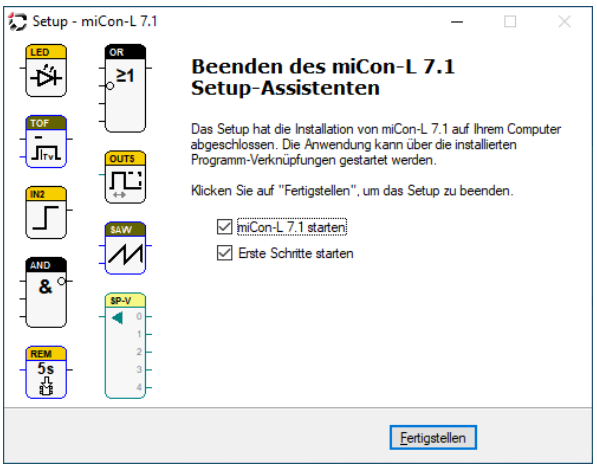


Abbildung 7: Fertigstellung der Installation

2.4 Software starten

Die Software miCon-L kann über den Startmenüeintrag geöffnet werden.

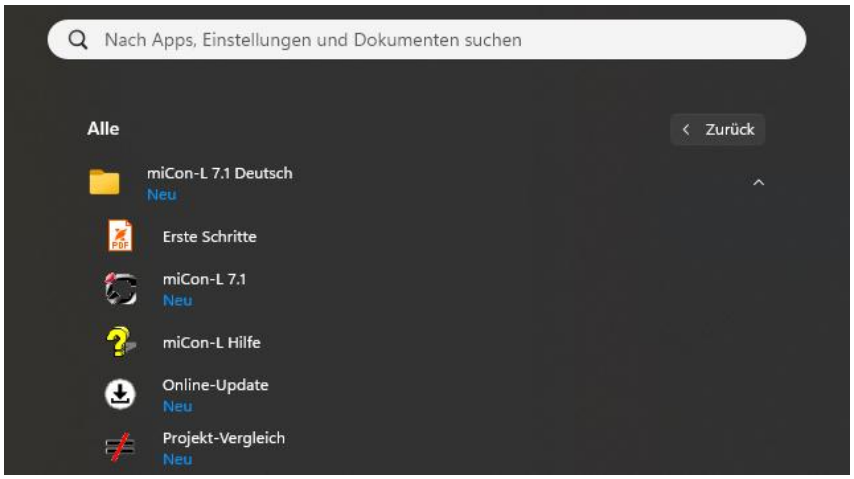


Abbildung 8: miCon-L im Startmenü

Der Start von miCon-L kann auch einfach über das Desktop-Symbol erfolgen.

Abbildung 9: miCon-L auf dem Desktop

Wurde während der Installation weder ein Startmenüeintrag noch ein Desktop-Symbol erstellt, so muss miCon-L direkt über das Dateisystem geöffnet werden.

.\miCon-L\BIN\miCon-L.exe

Nach dem Start von miCon-L kann ein vorhandenes (Beispiel-)Projekt geöffnet oder auch ein neues Projekt erstellt werden.

Die Erstellung eines neuen Projektes wird durch einen Dialog unterstützt, in dem Projektvorlagen für die unterstützten Steuerungstypen angeboten werden. Für einige Steuerungen gibt es auch Schnellstart-Vorlagen, die bereits E/A-Bausteine enthalten.

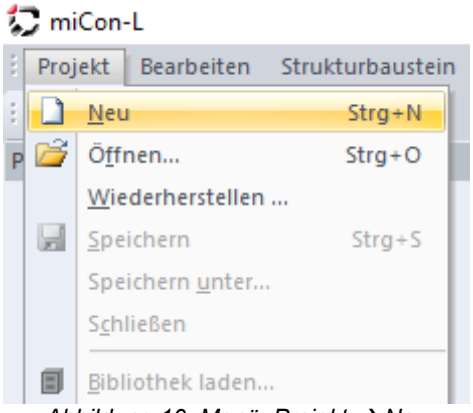


Abbildung 10: Menü: Projekt → Neu

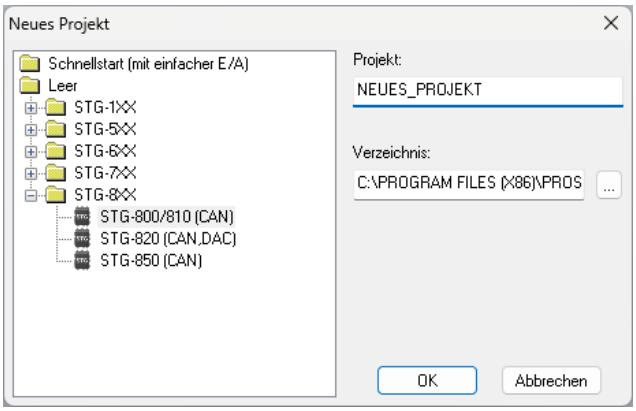


Abbildung 11: Dialog: Neues Projekt

2.5 mini-SPS an den Computer anschließen

Damit die mini-SPS vom Computer gefunden werden kann, muss diese korrekt mit Strom versorgt und das passende Kommunikationskabel korrekt angeschlossen werden.

Die Stromversorgung ist bei den mini-SPS auf 7 V – 32 V Gleichspannung ausgelegt, um 12V/24V-Systeme zu unterstützen.

Folgende Kommunikationskabel kommen je nach Steuerung zur Anwendung:

VK-10 (USB/RS232)	FTDI-Chip	STG-500
VK-12 (USB)	FTDI-Chip	STG-115/600
VK-16 (USB/TTL232)	FTDI-Chip	STG-550/570/580/650/680/700/8XX
VK-46 (USB/(TTL/RS)232)	ST-Chip	STG-500/550/570/580/650/680/700/8XX

Das VK-46 ist ein neues Kabel, das die Funktion von VK-10 und VK-16 vereint.

Die Kommunikation läuft immer über eine serielle Schnittstelle, die als native RS232 oder TTL232 implementiert ist und meist als virtueller COM-Port zur Verfügung gestellt wird.

Die Treiberinstallation für den virtuellen COM-Port erfolgt automatisch durch Windows. In Ausnahmefällen muss die Installation manuell erfolgen.

Setup für FTDI-Chip: .\miCon-L\SETUP\USBdriver

Setup für ST-Chip: STSW-LINK009 (Download von ST-Website, Registrierung erforderlich)

Nach Einrichtung der Kommunikationsstrecke muss im miCon-L-Projekt der korrekte COM-Port-Parameter eingestellt werden ( ).

→ COM64 ist der maximale COM-Port!

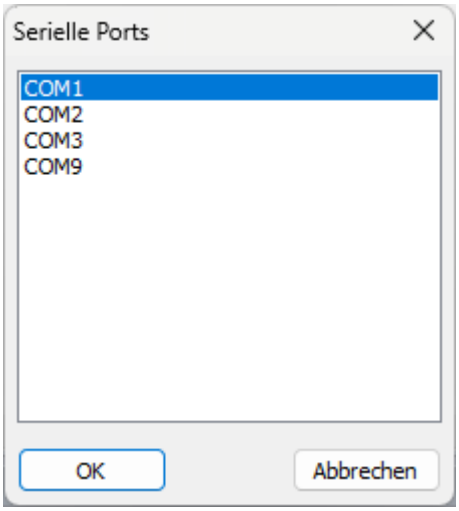


Abbildung 12: COM-Port Konfiguration

Der eingestellte COM-Port wird automatisch verwendet, wenn versucht wird, sich mit der Steuerung zu verbinden, z.B. um einen Download durchzuführen ( ).

Ist das Projekt bereits in die Steuerung geladen worden, so kann die Onlinebeobachtung direkt über das Online/Offline-Symbol (/) gestartet bzw. beendet werden.

Unabhängig von der Hardware können Projekte auch simuliert werden.

Für diesen Fall stellt miCon-L ein PC-Zielsystem zur Verfügung, um das Programm auch ohne reale Hardware abzuarbeiten. Die Simulation kann über ein Symbol in der Symbolleiste gestartet ( ) bzw. beendet ( ) werden.

2.6 Beispiel mit STG-600

Folgen Sie zuerst den Anweisungen in Kapitel 2.4 Software starten und legen Sie ein neues Projekt für die STG-600 an. Anschließend muss die Steuerung korrekt an den Computer angeschlossen werden. Folgen Sie hierfür den Anweisungen in Kapitel 2.5 mini-SPS an den Computer anschließen.

In diesem Beispiel ist die mini-SPS mit einem Potentiometer, einem Taster und 3 LEDs verbunden.

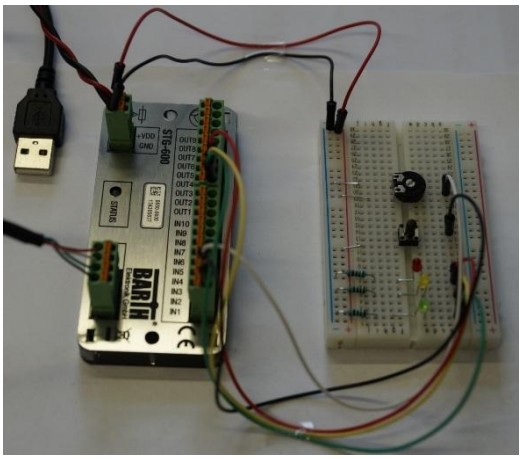


Abbildung 13: Versuchsaufbau (STG-600)

Die Ein- und Ausgänge an der Hardware sind hier wie folgt belegt:

IN1	Taster
IN3	Potentiometer
OUT1	Grüne LED
OUT2	Gelbe LED
OUT3	Rote LED

Bei gedrücktem Taster (IN1) sollen alle 3 LEDs eingeschaltet sein. Die Spannung am Mittelabgriff des Potentiometers soll einfach nur eingelesen und visualisiert werden.

Schritt 1: Zugriff auf das Potentiometer programmieren:

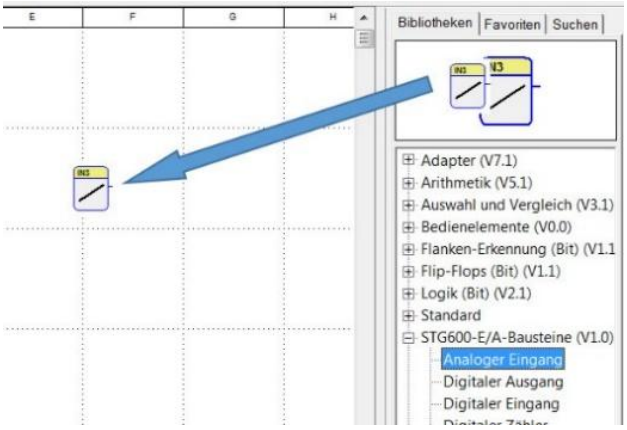


Abbildung 14: Baustein „Analoger Eingang“ einfügen

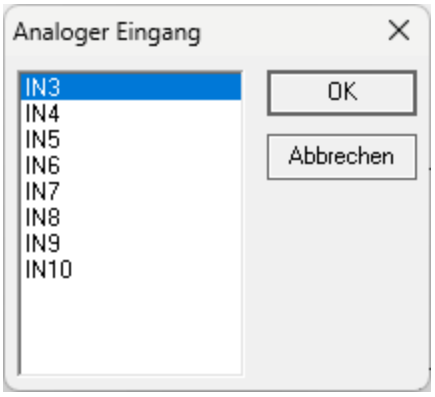


Abbildung 15: Parametrierung auf IN3

Dazu wird der Funktionsbaustein „Analoger Eingang“ per Drag-and-Drop direkt aus dem Bibliotheksbaum (oder aus dem Bausteinfenster) auf das Arbeitsblatt gezogen. Im sich öffnenden Parameterdialog wird der korrekte analoge Eingang gewählt.

Anschließend wird noch der Visualisierungsbaustein „Numerische Anzeige“ eingefügt, mit dem eine Onlinebeobachtung des angeschlossenen analogen Signals möglich ist.

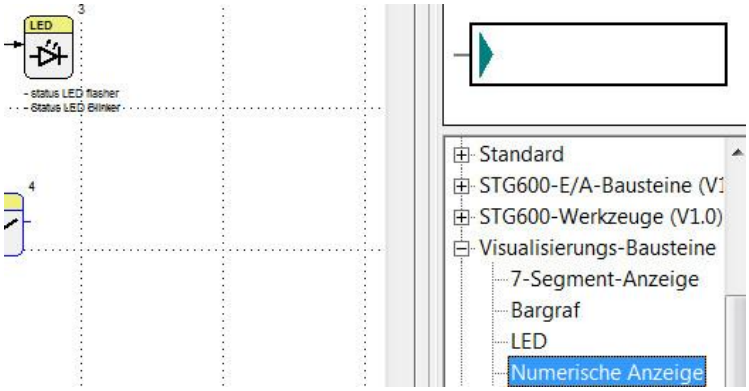


Abbildung 16: Baustein „Numerische Anzeige“ einfügen

Die Verbindung zwischen Ein- und Ausgang wird durch Linksklick auf den Ausgang des analogen Eingangs und erneutem Klick auf den Eingang der numerischen Anzeige erstellt.

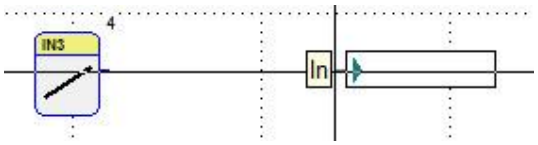


Abbildung 17: Anzeige mit Signal verbinden

Schritt 2: Programmieren der LEDs:

Im Folgenden müssen noch 4 weitere Funktionsbausteine auf dem Arbeitsblatt eingefügt, parametriert und miteinander verbunden werden.

Zuerst der digitale Eingang IN1, an dem der Taster angeschlossen ist:

Abbildung 18: Baustein „Digitaler Eingang“ auswählen

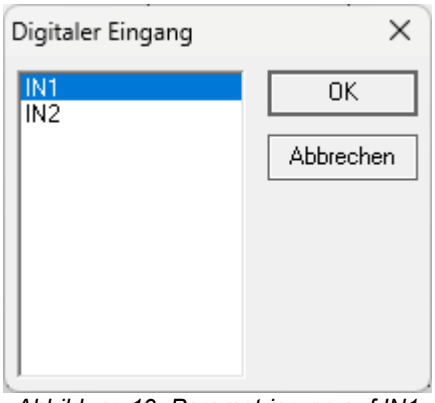


Abbildung 19: Parametrierung auf IN1

Anschließend werden die 3 Funktionsbausteine für die digitalen Ausgänge (OUT1/2/3), an denen die LEDs angeschlossen sind, eingefügt und direkt mit IN1 (Taster) verbunden.

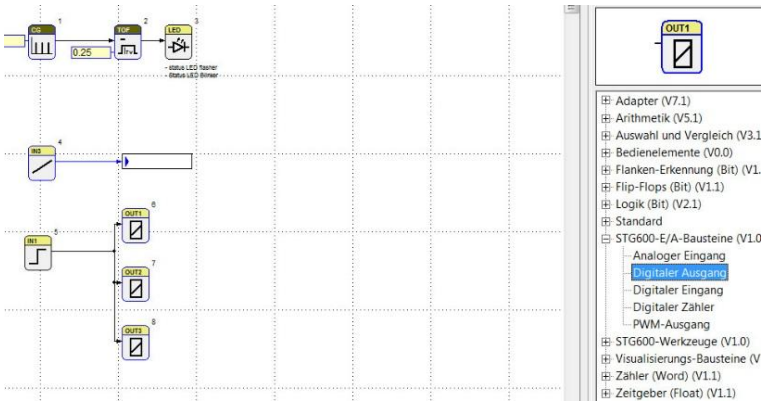


Abbildung 20: Fertiges Programm

Das fertige Programm wird per Download () auf die mini-SPS übertragen.

Beim Betätigen des „Download“-Symbols wird automatisch der Editiermodus verlassen, in den Inbetriebnahmemodus gewechselt, eine Verbindung zum Zielsystem aufgebaut und das Projekt auf das Zielsystem übertragen.

Im Onlinemodus sind die beiden blinkenden Funktionsblöcke Ausschaltverzögerung und Status-LED zu erkennen. Ein grünes Symbol zeigt den Zustand HIGH (TRUE) an. Beim Drehen am Potentiometer sind die sich ändernden Spannungswerte in der numerischen Anzeige zu erkennen.

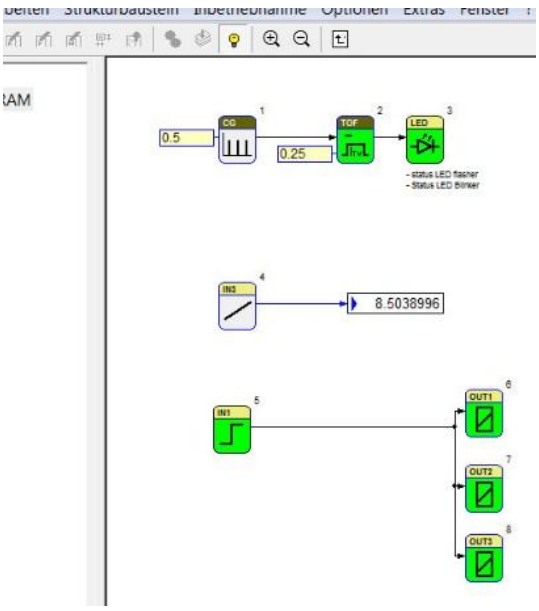


Abbildung 21: miCon-L im Online-Modus

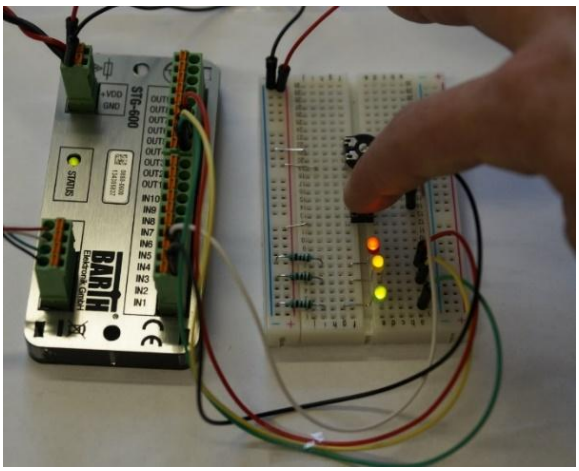


Abbildung 22: Versuchsaufbau (STG-600)

Beim Betätigen des Tasters leuchten nun die drei physischen LEDs und die entsprechenden Bausteine im miCon-L-Projekt.

3 Umsteiger

3.1 Was ist anders bzw. neu in miCon-L 7.1?

a) Installation

miCon-L 7.1 besitzt eine vollwertige Installation, im Gegensatz zum selbstentpackenden Archiv bei miCon-L 3.x. Die Installation meldet die von miCon-L benötigten Zugriffsrechte beim Betriebssystem (Windows) an, kann einen Startmenüeintrag und ein Desktopsymbol erzeugen, erfordert aber das grundsätzliche Recht, Software installieren zu dürfen.

b) Kommunikationseinstellungen

miCon-L 7.1 besitzt keine StartMe.exe mehr, wodurch sich für einige Steuerungen die COM-Port-Einstellung ändert. Diese wurde vereinheitlicht und verhält sich nun so wie bei den STG-8XX, d.h. der COM-Port wird für jedes Projekt einzeln, über die Werkzeugleiste () oder das Extras-Menü (siehe Kapitel 2.5 mini-SPS an den Computer anschließen), eingestellt.

c) Erste Schritte / FirstSteps (dieses Dokument)

Das „Erste Schritte“-Dokument wird nicht mehr über die StartMe.exe aufgerufen, weil es diese nicht mehr gibt, sondern direkt über das Hilfe-Menü in miCon-L. Die Ersten Schritte enthalten nun deutlich mehr Informationen und sind auch als zentrales Dokument gedacht, um verschiedene Informationsquellen miteinander zu verbinden und einen besseren Gesamtüberblick zu bieten.

d) Unterstützte Steuerungen

miCon-L 7.1 unterstützt weniger Steuerungen als miCon-L 3.8.1 direkt. D.h. es können für einige Steuerungen keine neuen Projekte mehr angelegt werden und es gibt auch die entsprechenden Beispielprojekte nicht mehr. Dies betrifft STG-32, STG-606, STG-860 und alle WCU-XXX. – Jedoch wird der Import für alle alten Projekte, und damit indirekt auch jede alte Steuerung, unterstützt (siehe nächstes Kapitel 3.2 Projektimport).

e) Verbesserungen und Neuerungen in miCon-L 7.1

- Die Anzahl der Bedienelemente hat während der Online-Parametrierung keinen Einfluss mehr auf die Reaktionsgeschwindigkeit, z. B. beim Betätigen eines Tasters.
- Im Editiermodus können Bausteine (Funktionsbausteine und Strukturbausteine) nun direkt per Drag-and-Drop aus dem jeweiligen Baum eingefügt werden.
- Die Positionierung der einzelnen Fenster (z.B. Projektbaum, Bibliotheksbaum, Meldungen) kann viel freier erfolgen und bietet deutlich mehr Flexibilität.
- Mögliche Bausteinreihenfolge-Probleme sind, durch eine automatische Hervorhebung der entsprechenden Bausteinnummer mit einem roten Kreis, deutlich leichter zu finden. Diese Funktion ist standardmäßig deaktiviert und kann manuell aktiviert werden. Dazu muss in Zeile 36 in der Datei .\BIN\icon.ini das führende Semikolon entfernt werden: `;checkblocksequence=display|dontprint`
- Es gibt einige neue Browser, um das Suchen von Signalen, Strukturbausteinanschlüssen, Fehlern, Texten, Bausteinen bzw. ganzen Bibliotheken und Makros zu vereinfachen.
- Es gibt weiterhin ein einzeln aktivierbares Suchfeld für Projektbaum und Bibliotheksbaum.
- Handling von Strukturbausteinen (Makros und Programmbausteine)
    - Die Anschlüsse von Strukturbausteinen (Input/Output) werden automatisch im Makrodesign platziert.
    - Im Makrodesign kann nun auch gescrollt werden (wichtig bei hohen Zoomstufen).
    - Die Datentypen von Inputs/Outputs können jetzt einfach über den Parameterdialog geändert werden (auch nach dem Einfügen auf dem Arbeitsblatt).
    - Mehr Dateiformate für Strukturbausteinbilder: BMP, EMF, PNG, JPG, TIF und GIF
    - Es gibt einen zusätzlichen Info-Text mit bis zu 200 Zeichen zur Kurzbeschreibung auf dem Symbol bzw. im Projektbaum.
- Die Variablenverwaltung wurde verbessert, so dass Variablen in Ordnern gruppiert werden können. Neu ist weiterhin, dass Variablen auch instanziierbar sein können.
- Es gibt einen zusätzlichen Standard-Funktionsbaustein „Rich-Text“, mit dem formatierte Texte (Farbe, Schriftart, ...) erstellt werden können (max. 30 kBytes im RTF-Format). Text, der z.B. mit Microsoft Word geschrieben wurde, kann auch direkt aus der Zwischenablage übernommen werden. (Bilder werden nicht unterstützt. Tabellen werden nur bedingt unterstützt.)
- Es gibt 2 neue Bedienelemente:
    - Link: Öffnen eines Dokuments oder einer Webseite
    - TODO-Mark: Erzeugen einer Meldung, einer Warnung oder eines Fehlers beim Download
- Online-Update: Ab miCon-L 7.1 kann die IDE selbständig nach Updates suchen.
- Projekt-Vergleich: Expertentool, um Projektunterschiede zu finden (nur Englisch).

3.2 Projektimport

Mit älteren miCon-L-Versionen erstellte Projekte können einfach mit der neuen Version 7.1 importiert und anschließend verwendet werden.

Beim Öffnen jedes Projektes prüft die IDE, ob ein Projektimport notwendig ist und zeigt gegebenenfalls den unten abgebildeten Dialog.

Während des Imports wird automatisch eine Kopie des alten Projektes erzeugt. Anschließend wird die Kopie an das neue miCon-L angepasst.

Das Verzeichnis für die Kopie kann im Import-Dialog geändert werden. Weiterhin sollten die Angaben nach Typ geprüft werden, ob die korrekte Steuerungsfamilie erkannt wurde.

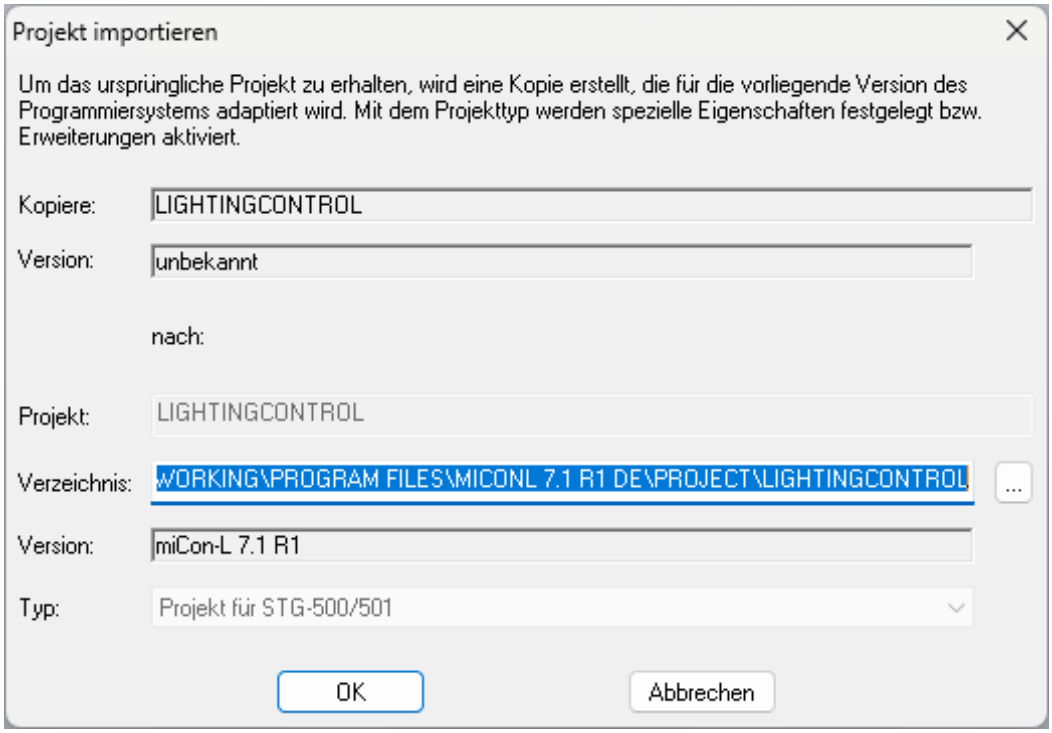


Abbildung 23: Dialog Projektimport

Der Projektimport ist, wegen erheblichen Unterschieden zwischen den verschiedenen Steuerungen, so konfiguriert, dass er den Projekttyp automatisch erkennt und nur Projekte gleichen bzw. kompatiblen Typs erzeugt, so dass der Nutzer nicht viel machen muss.

Für Projekte, deren Projekttyp nicht automatisch ermittelt werden kann, wird STG-850 als Default-Projekttyp verwendet.

4 Mehr Informationen

4.1 Wo finde ich weitere Informationen?

Weitere Informationen sind im Internet auf den folgenden **Webseiten** verfügbar:

- Firma BARTH® Elektronik GmbH: www.barth-elektronik.com
- miCon-L-Produktseite: https://barth-elektronik.com/miCon-L/

Erklärungen zu miCon-L sind in der **„Online-Hilfe“** der IDE verfügbar. (F1-Taste nutzen)

Erklärungen zu den Funktionsbausteinen sind für jede Bibliothek (und jeden Baustein) in einer separaten **„Baustein-Hilfe“** verfügbar. (Aufruf über das Kontextmenü des Bausteins)

Dieses Dokument (**FirstSteps**) wird während der Installation im Startmenü verlinkt und ist jederzeit über das „**?**“-Symbol im Menü von miCon-L aufrufbar. Weiterhin ist es online verfügbar.

Kaufmännischer Support: BARTH® Elektronik GmbH → office@barth-elektronik.de
Technischer Support: BARTH® Elektronik GmbH → support@barth-elektronik.de
Nutzerfeedback: BARTH® Elektronik GmbH → info@barth-elektronik.de

Auf das Thema **Beispiele** wird **im nächsten Kapitel** eingegangen.

4.2 Beispielprojekte

Beispielprojekte sind im miCon-L-Installationsordner zu finden:

.\miCon-L\PROJECT\SAMPLE_PROGS\

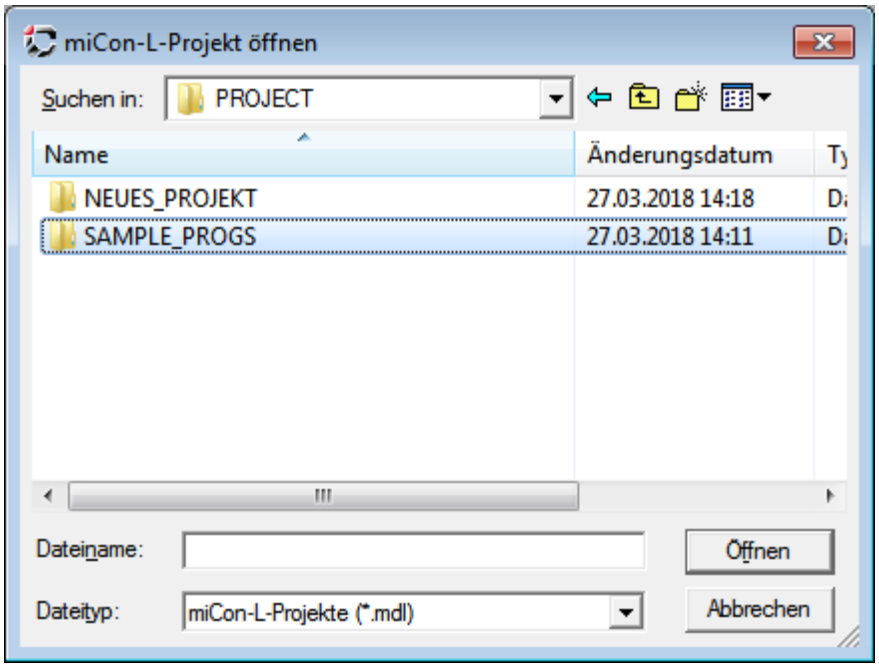


Abbildung 24: Dialog Projekt öffnen (SAMPLE_PROGS)

Es gibt **allgemeine Beispiele**, die hauptsächlich für die Simulation als Zielsystem gedacht sind, jedoch auch in alle STG-8XX downloadbar sind. – Mit diesen allgemeinen Beispielen werden grundsätzliche Prinzipien bzw. Lösungen für verschiedene kleine Aufgaben gezeigt.

Weiterhin gibt es für jede Steuerung **Beispiele**, die sich nur auf die spezifischen Bausteine beziehen.

4.3 Gut zu wissen / Tipps

a) Motivation

Für einfache Aufgaben ist ein tiefes Verständnis nicht notwendig und miCon-L bietet einen sehr schnellen und einfachen Einstieg in die SPS-Welt.

Werden Aufgaben komplexer oder die Anforderungen an die Echtzeit strenger, und dass bei gleichzeitig geringer Entwicklungszeit, dann ist für die Lösung oft ein sehr viel tieferes Verständnis auf vielen Ebenen notwendig!
Das notwendige Verständnis betrifft mehrere Gebiete:

- Funktionsweise und Bedienung von miCon-L (Editor)
- Funktionsweise und Anwendung der verschiedenen Funktionsbausteine
- Eigenschaften der Laufzeitumgebung (Funktionsumfang, Performance, Verhalten, ...)
- Eigenschaften der Hardware (Schnittstellenverhalten, Speichereigenschaften, …)
- Feldbusse, Sensorik, Aktorik und weitere angeschlossene Geräte
- Signalverarbeitung und Steuerungs-u. Regelungstechnik

Die Liste lässt sich je nach Aufgabengebiet noch fortsetzen, soll aber dennoch verdeutlichen, dass komplexe Lösungen trotz guter Hilfsmittel einen erheblichen Aufwand bedeuten können.

ProSign möchte einige Best Practices empfehlen:

- Persönliche Grundeinstellung: Learning by Doing → Nutzung der Simulation!
- Entwicklung in kleinen Schritten → Zerlegung in beherrschbare Teilaufgaben
- Beachten der Bausteinreihenfolge
- Trennen von Eingabe, Verarbeitung und Ausgabe, indem z.B. die Verarbeitung weitestgehend in Makros mit klaren Schnittstellen gekapselt wird. (Variablen als verborgene Schnittstellen sollten nur in Ausnahmefällen genutzt werden!)
- Selbsterklärende Programme:
    - Begrenzung der Anzahl der Bausteine auf einem Arbeitsblatt (z.B. 30)
    - Sinnvolle Namen für Makros, Schnittstellen (Input/Output) und Variablen
    - Text-Kommentare und Rich-Text zur Dokumentation verwenden

Die Simulation zusammen mit der Online-Beobachtbarkeit bietet sehr gute Voraussetzungen, um Teilprogramme frühzeitig und in kleinen Schritten zu entwickeln und zu verifizieren.

b) Konzepte der Bausteine und Bibliotheken

Die Bausteine und Bibliotheken sind **historisch gewachsen** und wurden durch verschiedene Entwickler umgesetzt. Es wurde stets versucht, eine große Stabilität zu wahren.

Bibliotheken gruppieren Bausteine ähnlicher Funktionsklassen. miCon-L unterstützt auch andere Gruppierung über das Konzept der Favoriten. **Favoriten** können in jedem Projekt erstellt, bearbeitet, exportiert und importiert werden.

Anordnung von Bausteineingängen: Eingänge befinden sich normalerweise nur auf der linken Seite können aber auch unten, z.B. als Steuersignale, angeordnet sein.

Anordnung von Bausteinausgängen: Ausgänge befinden sich normalerweise nur auf der rechten Seite, können aber auch oben, z.B. als Anzeigesignale, angeordnet sein.

Logische Verarbeitung bei einfachen Bausteinen:

[INPUT1] [OPERATION] [INPUT2] → [OUTPUT]

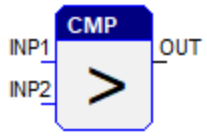


Abbildung 25: Vergleicher: größer

INP1 > INP2 → OUT

INPUT1	... **INP1**	(links oben)
OPERATION	… **>**	(Bausteinsymbol)
INPUT2	… **INP2**	(links unten)
OUTPUT	… **OUT**	(rechts oben)

Farbe von Bausteinumrandung und Pin zeigt den Datentyp an (**blau:** FLOAT, **schwarz**: BIT)

c) Instanziierung

Für die meisten Anwendungen werden nur globale Parameter und globale Variablen benötigt. Nur in Ausnahmefällen werden Makrobausteine mehrfach und mit unterschiedlichen Parametern im Projekt verwendet. Dann müssen die Randbedingungen des Kapitels "Klassen und Instanzen" in der miCon-L-Hilfe beachtet werden.

Es gibt einige wenige Funktionsbausteine, die keine globalen, sondern instanziierbare (lokale), Bausteinparameter nutzen. Das betrifft die Bedienelemente Schalter / Taster und Schieberegler und weiterhin die E/A-Bausteine mit Parameterdialog. In diesen Fällen ist die Empfehlung Parameteränderungen stets in der Klasse (Editiermodus) UND der Instanz (Inbetriebnahmemodus) durchzuführen.

d) Was macht den miCon-L-Alltag leichter?

- Bei Fragen zuerst in der Bausteinhilfe, der miCon-L-Hilfe oder den FirstSteps schauen.
- Copy und Paste verwenden (Strg + C, Strg + V)
- Export und Import verwenden → Das Rad nicht zweimal erfinden!
- Gerade im Supportfall sind möglichst klare Anforderungen wichtig.
- Wenn Projekte weitergegeben werden (bitte das gesamte Projektverzeichnis!), dann als zip-Datei gepackt. Der Ordner RESTORE kann aus dem Archiv gelöscht werden.

4.4 Wie funktioniert das Technologiepaket miCon-L?

Das Technologiepaket miCon-L besteht aus dem Block-Diagramm-Editor (miCon-L-IDE auf dem PC) und verschiedenen Zielsystemen (STG-XXX mit miCon-L-Laufzeitumgebung).

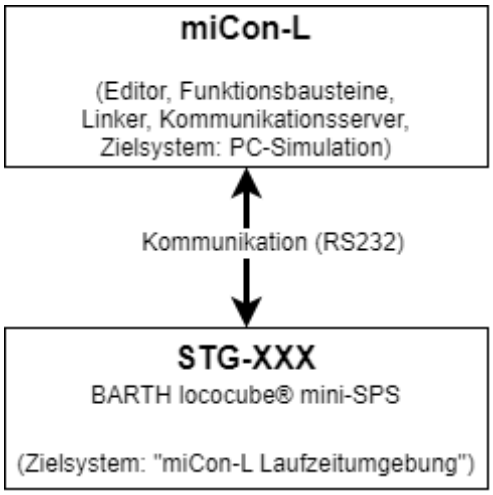


Abbildung 26: miCon-L Kontext

Mit dem Block-Diagramm-Editor kann im Editiermodus ein grafisches Programm (Anwendungsdesign) erstellt werden, das hierarchisch organisiert ist. Dabei können unter anderem Funktionsbausteine eingefügt, Strukturbausteine (Makros) erstellt und eingefügt und Verbindungslinien gezogen werden.

Abbildung 27: Zustände des Editors

Beim Wechsel in die Inbetriebnahme erfolgen die Instanziierung und Adressvergabe. D.h. für alle Daten und Parameter der verwendeten Bausteine und Makros wird Speicher allokiert

Die Adressen des jeweiligen Speichers werden über logische Namen verwaltet. Speicher gleichen Datentyps werden zu Blöcken zusammengefasst, die im Kontext von miCon-L Ressourcen genannt werden. (Eine Ressource ist also ein benannter Speicherblock mit einer bestimmten Anzahl von Elementen gleicher Größe.) Typische Beispiele für logische Adressen innerhalb der Ressourcen sind M1.0, M2.7, DF1, DF2, DW0, DW1, PF1, PF2.

Das Nullelement (M0.0, DF0, DW0, PF0, …) ist immer vorhanden und mit 0 initialisiert und wird automatisch verwendet, wenn z.B. Eingänge von Makros oder Bausteinen nicht beschaltet sind. Alle offenen Eingänge eines Datentyps nutzen das gleiche Nullelement, d.h. Änderungen an einem Eingang wirken sich auf alle aus.

Beim Download in das Zielsystem durchläuft das grafische Programm zwei Transformationsschritte.

Im ersten Schritt wird die hierarchische Struktur aufgebrochen und in eine große lineare Liste überführt und in Textform ausgegeben (Q-Datei im TEMP-Verzeichnis). Dieser Zwischencode ist normaler ASCII-Text und enthält alle Bausteinaufrufe und alle genutzten logischen Adressen. – Auf Basis dieses Zwischencodes ermittelt die IDE den notwendigen Gesamtspeicherbedarf je Ressource.
Es wird eine Ressourcenanforderungsliste an das Zielsystem gesendet, welches mit einer gültigen Gerätebeschreibungsdatei antwortet, falls genug Speicher (RAM, Flash) zur Verfügung steht, andernfalls mit einer Fehlermeldung. – Die Gerätebeschreibungsdatei enthält die physischen Adressen der Bausteinfunktionen und die physischen Adressen für die verschiedenen Speicherbereiche.
Im zweiten Schritt wird nun der Zwischencode (Q-Datei) mit Hilfe des Linkers und der Gerätebeschreibungsdatei übersetzt. Dabei werden die logischen Adressen gegen physische Adressen ausgetauscht. Das Ergebnis des Linkens ist eine Modul-Verbindungs-Liste (MVL), die in binärer Form vorliegt.
Diese binäre MVL und die Parameter aus dem Projekt werden als sogenannte Wiederanlaufdaten in die Steuerung geladen und dort nicht-flüchtig abgespeichert.
Die Laufzeitumgebung kann diesen Konfigurationsdatensatz (Wiederanlaufdaten) verarbeiten. (Der Block-Diagramm-Editor hat keine Informationen über den Inhalt der Funktionsbausteine, da diese fertig ausprogrammiert und kompiliert in der Laufzeitumgebung enthalten sind. Der Editor erzeugt also wirklich nur eine Aufrufliste der Zielfunktionen und stellt die jeweiligen Parameter bereit.)

Die Verarbeitung im Zielsystem wird durch den Dispatcher gesteuert, der die MVL liest und die entsprechenden Zielfunktionen aufruft. Nachdem eine Zielfunktion abgearbeitet wurde, wird die nächste Zielfunktion (der nächste Funktionsbaustein) aufgerufen.
Die MVL wird zyklisch, mit der eingestellten Taskzykluszeit, abgearbeitet. – Der letzte Baustein in der MVL teilt dem Dispatcher mit, dass ein vollständiger Durchlauf fertig ist. Der Dispatcher beendet daraufhin seine Abarbeitungsschleife. Ein neuer Dispatcher-Aufruf wird über ein Timer-Modul im Takt der Taskzykluszeit gestartet.

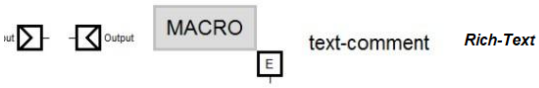
Über die Kommunikationsverbindung kann miCon-L lesend und schreibend auf den Speicher in der Steuerung zugreifen.
Der lesende Zugriff wird für die Online-Beobachtung des Zielsystems benötigt und der schreibende Zugriff für die Online-Parametrierung.
Werte, die während der Online-Parametrierung im flüchtigen RAM geändert wurden, sind nach dem nächsten Start des Zielsystems nicht mehr vorhanden. Es muss ein neuer Download durchgeführt werden, um die Wiederanlaufdaten zu aktualisieren.

4.5 Bausteinübersicht

miCon-L enthält viele verschiedene Bausteinbibliotheken, in denen jeweils Bausteine eines bestimmten Typs zusammengefasst sind. – Auf Grund ihrer Verschiedenheit unterstützt keine Steuerung alle Bausteine, da es sehr viele spezifische Bausteine gibt (z.B. E/A-Bausteine und Schnittstellenbausteine).
Es gibt jedoch auch „Basis-Bibliotheken“, ohne Abhängigkeiten von physischen Schnittstellen, die für alle Steuerungen verfügbar sind (Logik, Arithmetik, Adapter, Zeitgeber, …).
Viele Bausteine sind statisch in ihrer Anwendung, d.h. die Eingänge und Ausgänge haben festgelegte Datentypen und Funktionen. Es gibt aber auch viele Bausteine, bei denen erst bei der Verwendung z.B. der Datentyp festgelegt wird (Arithmetik, Adapter, …) oder die Anzahl der Anschlüsse (Logik, CAN senden/empfangen).

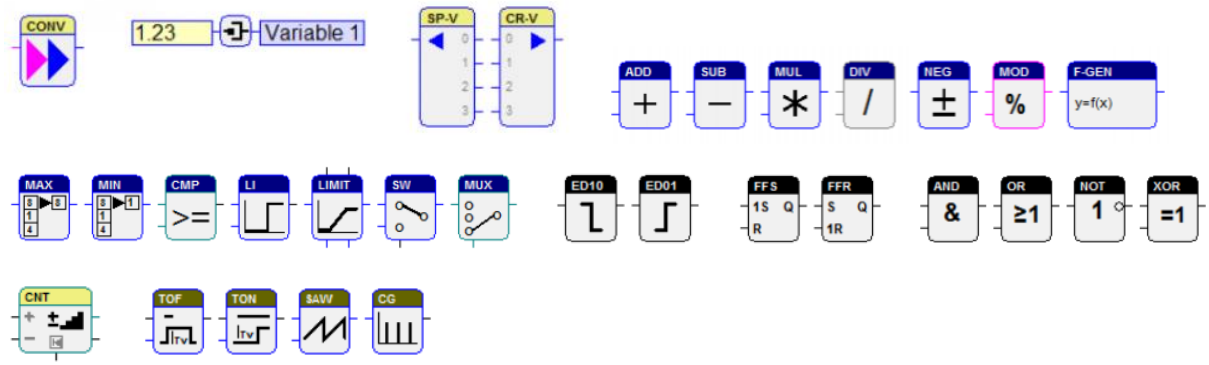
Grundsätzlich hat jeder Funktionsbaustein eine eigene Hilfe, die im Bibliotheksbaum oder auf dem Arbeitsblatt über das Kontextmenü geöffnet werden kann.
Der folgende Abschnitt gibt einen kurzen Überblick über verschiedene Bausteingruppen.

Standardbausteine (Input, Output, Enable, Text-Kommentar, Rich-Text)



- Input und Output werden benötigt, um die Schnittstellen von Makros zu definieren
- Enable wird benötigt, um den Aufruf von Makros zur Laufzeit zu steuern
- Text-Kommentar und Rich-Text dienen der besseren Dokumentation des Programms

Basisbausteine (Adapter, Arithmetik, Auswahl und Vergleich, Flankenerkennung, Flip-Flops, Logik, Zähler, Zeitgeber)

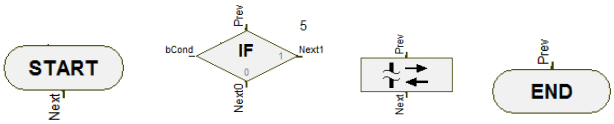


- Die Basisbausteine werden selbst für einfache Aufgaben in einer großen Breite benötigt.

Erweiterte Basisbausteine (Numerik, Regler, Schieben und Rotieren, Signalgeneratoren, Signalverarbeitung, Standardübertragungsglieder)

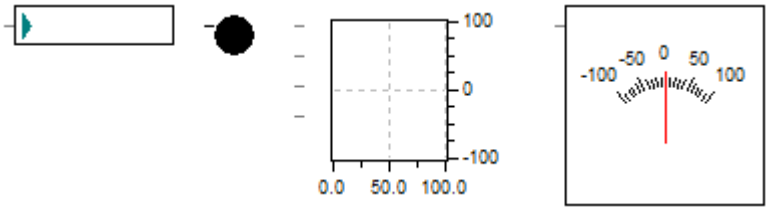
- Diese Bausteine stellen im Wesentlichen weitere Mathematikfunktionen und Funktionen zur Signalverarbeitung bereit und sind für die STG-8XX nutzbar.

Flow-Chart-Bausteine (START, IF, STEP, END)



- Mit den Bausteinen können ablauforientierte Programme realisiert werden.
  → Programme ohne Flow-Chart sind datenflussorientiert (d.h. strikte Bausteinreihenfolge)
- **Beachte**: Flow-Chart-Bausteine können nicht so einfach mit anderen kombiniert werden.
  → Siehe: Kapitel **Einführung** in der Bausteinhilfe für die Flow-Chart-Bibliothek

Visualisierungsbausteine (Numerische Anzeige, LED, Trend, Zeiger-Instrument, …)



- Die Visualisierungsbausteine dienen der Onlinebeobachtung des Zielsystems.
- Die Bausteine haben keine eigene Zielfunktion in der Steuerung und belasten das System nur bei aktiver Online-Kommunikation.

Bedienelemente (Schalter / Taster, Schieberegler, …)

- Die Bedienelemente dienen der Online-Parametrierung des Zielsystems.
- Die Bausteine haben keine eigene Zielfunktion in der Steuerung und belasten das System nur bei Wertänderung während aktiver Online-Kommunikation.
- Der Link-Baustein und der Baustein TODO-Markierung sind nicht für die Online-Parametrierung vorgesehen, unterstützen aber bei der Dokumentation, in der Inbetriebnahme und beim Download.

Hardwarespezifische Bausteine (E/A-Bausteine, Kommunikationsschnittstellen, …)

- Jede Steuerung hat spezifische Hardwareschnittstellen, die direkt durch eine oder mehrere eigene Bibliotheken unterstützt werden.

4.6 Glossar und Datentypen

Tabelle 1: Glossar - Allgemeines

Begriff	**Erklärung**
Applikation / Anwendungsdesign	Grafisches Programm (Anwenderprogramm) oder einfach nur „Programm“
Abarbeitungsreihenfolge / Bausteinreihenfolge	Reihenfolge, in der Bausteine abgearbeitet werden. Die normale Programmierung erfolgt datenflussorientiert, und die Reihenfolge wird über die Bausteinnummer vorgegeben (Start ist bei Bausteinnummer 1).
Abarbeitungszeit	Zeit, die für die Bearbeitung des gesamten Programmbausteins benötigt wird (je nach Applikation und Zielsystem in der Größenordnung von wenigen Mikrosekunden bis viele (100) Millisekunden).
Baustein	Synonym für Funktionsbaustein und Strukturbaustein
Bausteinbibliothek / Bibliothek	Strukturierungsmittel für Funktionsbausteine Bibliotheken können geladen, entfernt und ersetzt werden.
Blockdiagramm	Grafische Darstellung der Automatisierungslösung mit Hilfe von Funktionsbausteinen und Makro
E/A	Eingänge und Ausgänge
Echtzeitfehler	Tritt auf, wenn die Abarbeitungszeit größer als die Taskzykluszeit des Programms ist. Kurzzeitige Echtzeitfehler führen meist nicht zu dauerhaften zeitlichen Problemen. – Häufige Echtzeitfehler sollten in jedem Fall vermieden werden, da einige Bausteine auf einen äquidistanten Aufruf angewiesen sind. Eine Reduktion bzw. Vermeidung ist durch eine größere Taskzykluszeit oder effizientere Programme möglich.
Funktionsbaustein	Der Funktionsbaustein wird in der SPS-Welt auch häufig Funktionsblock genannt. In der Programmiersoftware miCon-L ist der Funktionsbaustein das zentrale Programmiermittel und stellt die kleinste verwendbare Einheit eines Anwenderprogramms dar.
IDE	Integrated Development Environment (wird synonym zu Block-Diagramm-Editor (also miCon-L) verwendet)
Kommunikationsserver / Onlineserver	Softwaremodul, das auf PC-Seite den Kommunikationstreiber bereitstellt und den grafischen Editor mit dem Zielsystem verbindet, so dass eine Datenkommunikation stattfinden kann.
Linker	Softwaremodul, das das grafische Programm mit logischen Adressen, in physische Adressen des Zielsystems übersetzt. Der direkte Input für den Linker ist der Zwischencode, der vom grafischen Editor erzeugt wurde.

Makro(-baustein)	Makrobausteine enthalten Teilfunktionen und dienen der Modularisierung. Sie sind nicht als klassisches Unterprogramm zu verstehen, sondern erzeugen bei jeder Verwendung den vollen Code.
Programmbaustein (Task)	Ein Programmbaustein wird in der Konfigurationsebene eingefügt. miCon-L unterstützt nur einen Programmbaustein, der bereits eingefügt ist. Innerhalb des Programmbausteins wird die eigentliche Funktionalität programmiert.
Projekt	Ein (miCon-L-)Projekt ist eine Sammlung von allen „konfigurierbaren“ Dateien, die notwendig sind, um eine Lösung für eine Steuerung vollständig zu unterstützen. Das eigentliche Anwendungsdesign (Modell) wird in der MDL-Datei gespeichert. Es gibt aber noch weitere Dateien, die z.B. die Variablen enthalten oder die Arbeitsbereich-Einstellungen (IWS-Datei, iCon-L Workspace Settings).
Simulation („Software-SPS“)	(Standard-)Simulation, die ein PC-Zielsystem zur Verfügung stellt, um das Anwendungsdesign auch ohne reale Hardware-SPS abzuarbeiten. Die Simulation ist ideal zur schnellen und unkomplizierten Prüfung der eigentlichen Programmlogik.
SPS	Speicherprogrammierbare Steuerung → BARTH lococube® mini-SPS Wird eingesetzt zur Steuerung oder Regelung einer Maschine oder Anlage und wird auf digitaler Basis (mit miCon-L) programmiert.
Strukturbaustein	Sammelbegriff für Makro- und Programmbausteine
Taskzykluszeit (Zykluszeit)	Zeit, nach der der Programmbaustein erneut abgearbeitet wird. (Die Programmabarbeitung erfolgt zyklisch, mit der Taskzykluszeit als Periodendauer.)
Verbindungslinie / (Signallinie)	Verbindungslinien dienen dem Datenaustausch zwischen Funktionsbausteinen und Strukturbausteinen. Normalerweise können nur Ausgänge mit Eingängen verbunden werden. Eine Verbindungslinie kann auch durch eine Variable ersetzt werden.
Zielfunktion	Eine Zielfunktion ist eine ausprogrammierte Funktion im Zielsystem, die eine Bausteinfunktionalität umsetzt.
Zielsystem / Laufzeitumgebung	Ein Zielsystem ist, im Kontext von miCon-L, eine Laufzeitumgebung, die ein miCon-L-Programm verarbeiten kann. Das Zielsystem enthält alle unterstützten Zielfunktionen.
Zwischencode (Q-Datei)	Der Zwischencode ist eine Übersetzung des grafischen hierarchischen Programms in eine lineare Darstellung in Form einer ASCII-Textdatei. z.B. .\miCon-L\TEMP\TMP001.Q

Tabelle 2: Glossar - Bedien-u. Datenkonzept

Begriff	**Erklärung**
ablauforientiert / steuerflussorientiert	Gegensatz zu datenflussorientiert Die Programmierung erfolgt mit Flow-Chart-Bausteinen. Der Steuerfluss bzw. Ablauf kann zur Laufzeit manipuliert werden.
Anschlussattribut (Attribut)	Bezeichnet Anschlussparameter an Signaleingängen und Variablen an Signaleingängen und Signalausgängen.
Bausteinparameter	Parameter, die innerhalb des Bausteins gespeichert sind und nicht über Anschlussattribute geändert werden können. Diese sind meist instanziierbar. z.B. Auswahl des „Kanals“ bei Ein-u. Ausgängen oder der Anzeigetext beim Bedienelement Schalter / Taster.
Daten(-speicher)	Datenspeicher ist flüchtiger RAM-Speicher, der in den meisten Fällen mit 0 vorinitialisiert ist. Datenspeicher ist sehr oft der limitierende Faktor bei „großen Projekten“ bzw. „kleinen Zielsystemen“. Jeder Bausteinausgang braucht Datenspeicher. Die Bausteineingänge verbrauchen keinen Datenspeicher, da sie die bereits allokierten Speicher der angeschlossenen Signale benutzen.
datenflussorientiert	Gegensatz zu ablauforientiert Die Abarbeitungsreihenfolge wird strikt durch die Bausteinnummer vorgegeben und kann nicht geändert werden. Es können maximal Teilfunktionen aktiviert oder deaktiviert werden (→ Makro mit Enable-Baustein).
Download	Download bezeichnet den Vorgang der Übertragung des transformierten Programms in das Zielsystem.
Editiermodus / Bearbeitungsmodus	Der Editiermodus ist der Grundzustand des Programmiersystems. In diesem Zustand erfolgt die hauptsächliche Bearbeitung des Projekts.
Flow Chart / Programmablaufplan	Flow Charts ermöglichen die Ablauf- bzw. die Steuerflussprogrammierung.
global (nicht instanziierbar)	Gegensatz zu lokal bzw. instanziierbar Attribute und Bausteinparameter, die global sind, sind im Editiermodus und im Inbetriebnahmemodus stets gleich.
GVR	Abkürzung für Globale Variablen und Referenzen GVR werden über einen Extra-Dialog konfiguriert und können Variablen mit verschiedenen Eigenschaften zur Verfügung stellen. Diese Informationen werden in einer separaten Projekt-Datei gespeichert und können exportiert und importiert werden.
Inbetriebnahme(-modus)	Der Inbetriebnahmemodus ist ein spezieller Modus zur Instanziierung und zur Erzeugung des Programms. In diesem Modus werden alle Speicher (logische Adressen) bereitgestellt. Instanziierbare bzw. lokale

	Attribute und Bausteinparameter können je Instanz zugewiesen werden.
Instanz	Eine Instanz ist eine Kopie / Ableitung von der Klasse eines „Bausteins“. Die Instanzen werden beim ersten Wechsel in den Inbetriebnahmemodus erzeugt. Dabei werden die Instanzwerte (Attribute und interne Bausteinparameter) mit den Klassenwerten initialisiert.
Klasse	Eine Klasse eines Attributs, Bausteins oder Strukturbausteins wird im Editiermodus erstellt und ist eine Art Vorlage für die abgeleiteten Instanzen.
Konfiguration	Im Kontext einer miCon-L-Applikation ist die Konfigurationsebene die oberste Ebene. Sie dient der grafischen Konfiguration. Im Inbetriebnahmemodus kann hier die Taskzykluszeit eingestellt werden. – miCon-L unterstützt nur eine Task (Programmbaustein).
Kontextmenü	Kontextmenüs werden in den Arbeitsblättern oder im Projekt- bzw. Bibliotheksbaum durch die rechte Maustaste aufgerufen. Die enthaltenen Befehle beziehen sich auf das Objekt unter dem Cursor.
lokal (instanziierbar)	Gegensatz zu global bzw. nicht instanziierbar Ein instanziierbares Signal (Attribut oder Bausteinparameter), ist nicht unbedingt in allen Systemzuständen des Block Diagramm Editors gleich. (Der Instanzwert kann vom Klassenwert abweichen.)
MVL	Modulverbindungsliste Lineare Repräsentation des Programms als Datensatz
Offline	Allgemeine Beschreibung dafür, dass keine aktive Kommunikation mit dem Zielsystem stattfindet.
Online(-modus)	Systemzustand, in dem miCon-L aktiv mit dem Zielsystem kommuniziert und Daten austauscht (mind. den Zykluszähler, der vom Editor als Zeitwert gezeigt wird).
Online-Beobachtung	Online-Beobachtung ist nur im Onlinemodus möglich. Dabei werden Daten aus dem Zielsystem gelesen. Dazu gehört der aktuelle Zykluszähler und die Daten für alle auf dem Arbeitsblatt vorhandenen Visualisierungsbausteine. Manche normalen Funktionsbausteine (Flip-Flops, Logik-Bausteine, …) visualisieren ebenfalls ihren Ausgangszustand.
Online-Kommunikation	Datenaustausch, der im Onlinemodus zwischen miCon-L und dem Zielsystem stattfindet. Eine hohe Kommunikationslast kann bei „schwachen“ Zielsystemen einen starken Einfluss auf das Echtzeitverhalten haben.
Online-Parametrierung	Online-Parametrierung ist nur im Onlinemodus möglich. Dabei werden Daten in das Zielsystem geschrieben (z.B. Werte auf Signallinien und Parameter).

Parameter	Parameter sind Bausteinparameter oder Anschlussparameter. Diese können entweder global oder lokal (instanziierbar) sein. Sie können als klassische Konstanten verstanden werden.
Parameter(-speicher)	Parameterspeicher ist ein gesondert behandelter RAM-Speicher, der mit den Parametern aus dem Programm vorinitialisiert wird. Beim Download werden die Werte in einem nichtflüchtigen Speicher abgelegt und beim Start des Systems aus diesem wieder hergestellt. Parameterspeicher wird für Parameter-Attribute benötigt. Von Parametern sollte nur gelesen werden.
Programm(-speicher)	Der Programmspeicher enthält die Modulverbindungsliste und wird bei kleinen Mikrocontrollern direkt in den Flash gelegt. Die Größe des verfügbaren Programmspeichers bestimmt maßgeblich die maximale Größe des Programms (neben Datenspeicher und Abarbeitungszeit (CPU-Geschwindigkeit)).
Projektimport	Durch die IDE unterstützter Prozess zum Öffnen eines Projektes, das mit einer anderen (älteren) Version des Programmiersystems erstellt wurde.
Variable	Variablen sind Anschlussattribute, die entweder global oder lokal (instanziierbar) sein können. Datenvariablen können gelesen und geschrieben werden. Parametervariablen sollten nicht genutzt werden.

Tabelle 3: Basisdatentypen

Basisdatentyp	**Wertebereich**	**Speicher [Byte] + Farbe**	**Abgeleitete Ressource**	**Beispiel**
BIT bool	TRUE / FALSE	1	M .. Daten B .. Parameter	M0.0, M1.0, M1.1 B1.0
BYTE bool	8x TRUE / FALSE (8x BIT)	8	M .. Daten B .. Parameter	M0, M1, M2 B1
UCHAR positive Ganzzahl	0 ... 255	1	UCHARDATA .. Daten UCHARPARA .. Parameter	UCHARDATA0, UCHARDATA1 UCHARPARA1
WORD Ganzzahl	-2^{15} ... 2^{15} - 1 -32768 ... 32767	2	DW .. Daten PW .. Parameter	DW0, DW1 PW1
LONG Ganzzahl	-2^{31} ... 2^{31} - 1 -2.147.483.648 ... 2.147.483.647	4	DL .. Daten PL .. Parameter	DL0, DL1 PL1
FLOAT (IEEE754) Gleitkommazahl	± 1.175494e-38 … ± 3.402823e+38	4	DF .. Daten PF .. Parameter	DF0, DF1 PF1

5 Historie

Tabelle 4: Dokumentenhistorie

Datum	Version	Änderungen
02/2012	-	Erstellung (gemeinsame deutsche und englische Version)
07/2015	-	Vollständige Überarbeitung im Rahmen von miCon-L 3.3 Auftrennung in eigenständige deutsche und englische Versionen
05/2016	-	Kleine Anpassungen im Rahmen von miCon-L 3.7.0
03/2018	-	Teilweise Überarbeitung im Rahmen von miCon-L 3.8.0 Ergänzung der Kapitel „Was ist miCon-L?“ und „Informationen, Download-Quellen und Beispielprojekte“
10.06.2025	2.0.0	Vollständige Überarbeitung im Rahmen von miCon-L 7.1 Referenzen (Abschaltung der miCon-L-Webseite und des Forums)