Einstieg in MikroForth

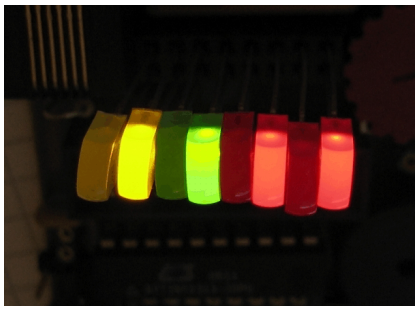

Abbildung 1

MikroForth ist ein FORTH-Compiler für den Attiny 2313. Damit ist gemeint: Das Programm wird auf einem PC eingegeben und kompiliert. Der dabei erzeugte HEX-Code wird anschließend mit einem Brennprogramm auf den Attiny übertragen.

Im Stile einer Schritt-für-Schritt-Anweisung wollen wir in diesem Kapitel zeigen, wie man mit MikroForth umgeht. Dazu werden wir mit MikroForth den Attiny so programmieren, dass er die Leuchtdioden an Port B in einem solchen Muster wie in Abb. 1 aufleuchten lässt. Die wenigen benötigten FORTH-Sprachelemente werden wir einfach angeben; wie sie funktionieren - und vor allem: wie man damit eigenständig Programme entwickelt, das werden wir dann in den folgenden Kapiteln darlegen.

Beginnen wir ganz vorne: mit der Installation von MikroForth. Kopieren Sie dazu einfach das Forth-Verzeichnis in das Programme-Verzeichnis (oder auch ein anderes Verzeichnis Ihrer Wahl). Öffnen Sie nun dieses Verzeichnis und starten Sie das Programm Forth2.

Während das Programm startet, lädt es die Datei “forthvoc.vok” mit dem Vorrat an Forth-Befehlen.

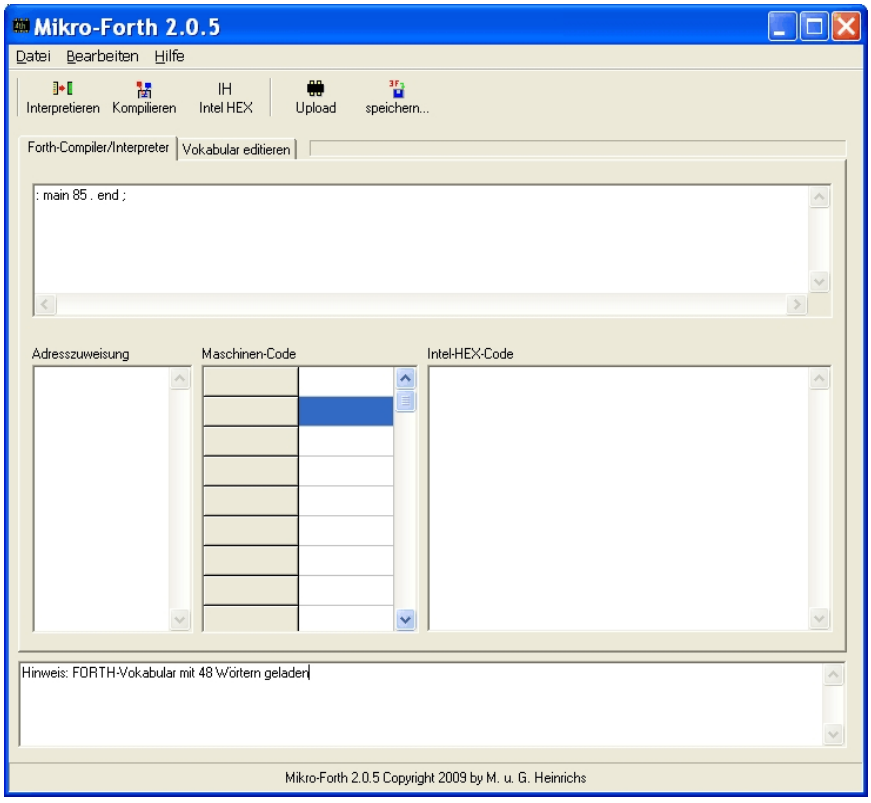

Abbildung 2

Unser erstes Programm besteht nur aus einer einzigen kurzen Zeile:

```
: main 85 . end ;
```

Geben Sie diesen Forth-Quelltext ganz oben im Formular ein (Abb. 2). Achten Sie auf die Leerzeichen zwischen den einzelnen Befehlen, insbesondere hinter dem Doppelpunkt und vor dem Semikolon; die Eingabe braucht man nicht mit der RETURN-Taste ↵ abschließen.

Unser Programm gibt zunächst die Zahl 85 am Port B aus; da die Zahl 85 im Zweiersystem als 01010101 geschrieben wird, sollte dies das gewünschte Muster bei den LEDs erzeugen (vgl. Abb. 1). Anschließend führt das Programm eine Endlosschleife aus.

Als nächstes betätigen wir die "Interpretieren"-Schaltfläche. Wir erhalten die folgende Warnung:

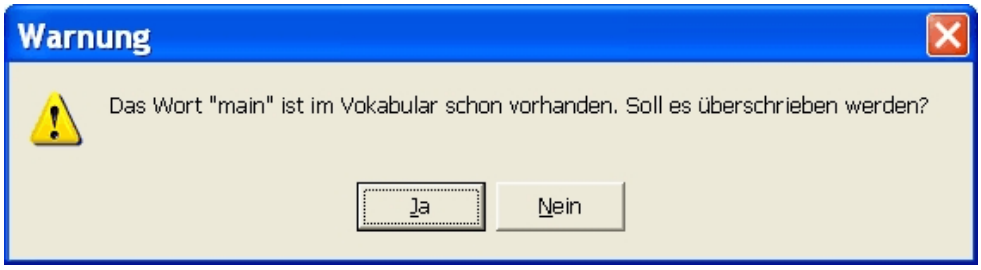

Abbildung 3

Was bedeutet das?

"." oder "`end`" sind sogenannte **Wörter**; diese stellen Befehle oder Befehlsfolgen dar, die der Mikrocontroller ausführen soll. Die Gesamtheit aller Wörter bezeichnen wir als **Forth-Vokabular**. Beim Interpretieren der eingegebenen Zeile wird hier dem Vokabular ein neues Wort "`main`" hinzugefügt, welches die Zahl 85 und die Wörter "." und "`end`" zusammenfasst. Offensichtlich existierte schon ein Wort "`main`" im Vokabular. Wie wir noch sehen werden, ist dieses Erstellen neuer Befehle ein wesentliches Konzept der Programmiersprache FORTH.

Da das alte "`main`"-Wort durch unser neues ersetzt werden soll, klicken wir bei dem Warnhinweis in Abb. 3 auf "Ja".

Aus diesem neuen Wort "`main`" muss nun Maschinencode für den Mikrocontroller erzeugt werden. Dazu betätigen wir die "Kompilieren"-Schaltfläche. Unser FORTH-Compiler besorgt sich aus der Datei "forthvoc.vok" die Programmschnipsel für die einzelnen Bestandteile von "`main`", also für die Wörter "." und "`end`" und fügt sie zu einem Gesamtprogramm zusammen. Im Adresszuweisungsbereich (Abb. 4) erkennt man die Zuweisung dieser Unterprogramme zu bestimmten Programmspeicheradressen; die Zeile "`. » $001A`" bedeutet z.B.: Das Unterprogramm für ".", welches für die Ausgabe auf Port B verantwortlich ist, beginnt bei der Adresse 26 = $1A. Das gesamte Programm wird dann im Maschinencodebereich angezeigt. Unter jeder Adresse finden wir ein Maschinencode-Wort, bestehend aus 2 Byte.

Im Logbuch entdecken wir weitere Einträge; auf deren Bedeutung werden wir zu einem späteren Zeitpunkt eingehen.

So wie es im Maschinencodebereich angezeigt wird, so wird das Programm auch im Mikrocontroller abgelegt (wenn man davon absieht, dass im Speicher des Mikrocontrollers das höherwertige Byte eines Maschinencode-Wortes nicht wie hier vor, sondern hinter dem niederwertigen steht). Die meisten Brennprogramme benutzen allerdings ein anderes Format, welches noch zusätzliche Kontrollbytes besitzt: das Intel-HEX-Format. Mit der Schaltfläche “Intel-HEX” wandeln wir deswegen nun unseren Maschinencode in dieses Format um. Der Intel-HEX-Code erscheint sogleich im Intel-HEX-Bereich (Abb. 4).

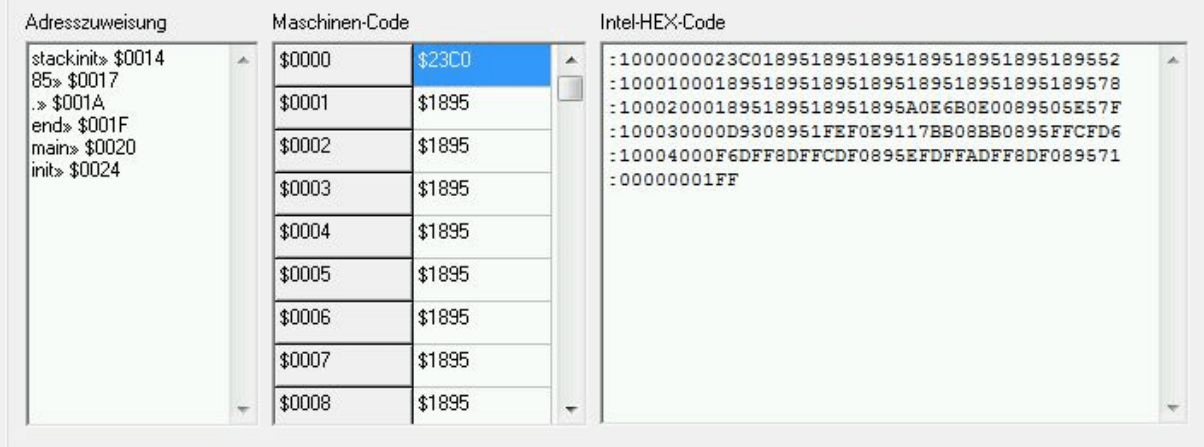

Abbildung 4

Zu guter letzt müssen wir den Code noch in den Attiny laden. Bei unserer Attiny-Platine kommt wieder das Uploader-Programm zum Einsatz. Wir rufen es auf, indem wir die “Upload”-Schaltfläche betätigen; der Intel-HEX-Code wird dann automatisch vom Uploader-Programm übernommen und kann auf dem üblichen Weg in den Attiny übertragen werden.

Und wenn Sie keinen Eingabefehler gemacht haben, die LEDs korrekt in die Buchsen gesteckt wurden, die Batterie noch voll ist und die Übertragung reibungslos funktioniert hat, ja - dann sollte auch das Bitmuster aus Abb. 1 tatsächlich angezeigt werden und wir gratulieren Ihnen zur erfolgreichen Implementation Ihres ersten FORTH-Programms!