
EasyLab Technical Paper

Manually programming the EasyKit Starter device

EasyKit Starter manuell programmieren

Simon Barner
barner@in.tum.de

Michael Geisinger
geisinge@in.tum.de

June 9, 2010

Note

This document contains information about the EasyLab software application. Please understand that as EasyLab is still under development, the information contained herein are subject to change without further notice.

Contents / Inhalt

1	Introduction / Einleitung	2
2	Disclaimer / Haftungsausschluss	2
3	Programming EasyKit Starter with <i>C</i> / EasyKit Starter mit <i>C</i> programmieren	3
3.1	Prerequisites / Voraussetzungen	3
3.2	Programming / Programmierung	3
3.3	Building the source / Den Quellcode übersetzen	3
3.4	Uploading the firmware / Übertragen der Firmware	4
4	Programming EasyKit Starter with <i>Assembler</i> / EasyKit Starter mit <i>Assembler</i> programmieren	5

1 Introduction / Einleitung

While EasyLab provides a convenient and efficient way to develop software for the EasyKit Starter device, it might be useful, e.g. for educational purposes, to program the device using the *C* or *Assembler* programming language.

The intention of this document is to provide a good starting point to access the EasyKit Starter device from *C* or *inline Assembler*, but it is neither a (low-level) *C/Assembler* programming tutorial nor a specification of the EasyKit Starter device or the software libraries used by the EasyLab code generator.

Obwohl EasyLab die angenehme und effiziente Entwicklung von Software für EasyKit Starter bietet, kann es beispielsweise für Zwecke der Ausbildung sinnvoll sein, das Gerät mit der Programmiersprache *C* oder *Assembler* zu programmieren.

Dieses Dokument soll einen guten Einstieg dafür bieten, wie EasyKit Starter mit *C* oder *Inline Assembler* programmiert werden kann, aber es kann weder eine (ausführliche) Einführung in die Programmiersprache *C* bzw. *Assemble* bieten noch ist es Spezifikation des EasyKit Starter oder der Software-Bibliotheken, die vom EasyLab-Codegenerator verwendet werden.

2 Disclaimer / Haftungsausschluss

Please note that

1. Programming the EasyKit Starter device with other means than with the EasyLab code generator is at your own risk – we are not responsible if the EasyKit Starter device or one of its peripherals become unusable.
2. There is no official support of any kind.

Beachten Sie Folgendes:

1. Die Programmierung von EasyKit Starter mit anderen Mitteln als mit dem EasyLab-Codegenerator erfolgt auf eigenes Risiko – wir können nicht dafür verantwortlich gemacht werden, wenn das Gerät oder eine der Peripheriekomponenten dadurch unbrauchbar wird.
2. Es erfolgt keine weitere offizielle Hilfestellung oder Unterstützung irgendeiner Art.

3 Programming EasyKit Starter with C / EasyKit Starter mit C programmieren

3.1 Prerequisites / Voraussetzungen

1. In order to manually program the EasyKit Starter, please ensure that you have installed EasyLab on your computer. Since EasyLab features a C code generator, it also provides all the required development tools.
2. Create a **copy** of the following folder, which will be your workspace (the path to EasyLab depends on where you have EasyLab installed):
1. Damit EasyKit Starter manuell programmiert werden kann, stellen Sie sicher, dass Sie EasyLab auf Ihrem Rechner installiert haben. Da EasyLab einen C-Codegenerator beinhaltet, bietet es auch alle benötigten Entwicklungswerkzeuge an.
2. Erstellen Sie eine **Kopie** des folgenden Ordners, der als Arbeitsumgebung dienen wird (der Pfad zur EasyLab-Installation ist davon abhängig, wo Sie EasyLab installiert haben):

`C:\Program Files\EasyLab\doc\easykit-starter-c\workspace`

3.2 Programming / Programmierung

You can now create your own C program using the `main.c` template in your copy of the workspace as a starting point. If you add new source (*.c) or header files (*.h), be sure to add them to the `SRCS` and `HDRS` section in `CMakeLists.txt` respectively.

Hint: The code generator uses an abstraction layer, that can also be used from a manually created C programs. See:

Sie können nun Ihr eigenes C-Programm schreiben, indem Sie die Vorlage `main.c` in Ihrer Arbeitsumgebung als Ausgangspunkt verwenden. Wenn Sie weitere Quelldateien (*.c) oder Headerdateien (*.h) hinzufügen, stellen Sie sicher, dass diese in der entsprechenden Sektion `SRCS` oder `HDRS` in der Datei `CMakeLists.txt` eingetragen sind.

Hinweis: Der Codegenerator verwendet eine Abstraktionsschicht, die auch aus selbst erstellten C-Programmen heraus verwendet werden kann. Siehe:

`C:\Program Files\EasyLab\runtime`

3.3 Building the source / Den Quellcode übersetzen

To build the source code, you need to perform the following steps:

Folgen Sie die Anweisungen, um den Quellcode zu übersetzen:

1. Open a command prompt (*Start* → *Run* → *cmd* → *OK*) and change to your workspace directory.
 2. Configure your source tree using CMake by running the `configure.bat` batch file. Please note that this is only needed once per workspace (in order to create the required makefiles).
 3. To build your C application, simply run the `build.bat` batch file. If you receive error messages, please ensure that the paths set up in `prologue.bat` and `CMakeLists.txt` (variable `EASYLAB_DIR`) match your system configuration.
Note: This will automatically re-run CMake in case you have modified the `CMakeLists.txt` file.
 4. Note: The build is performed in the `build` directory that contains all temporary files and that can safely be deleted if needed.
1. Öffnen Sie eine Eingabeaufforderung (*Start* → *Ausführen* → *cmd* → *OK*) und wechseln Sie in das Verzeichnis Ihrer Arbeitsumgebung.
 2. Konfigurieren Sie den Quellverzeichnisbaum mit CMake durch das Ausführen von `configure.bat`. Bitte beachten Sie, dass dies nur einmal je Arbeitsumgebung nötig ist (um die benötigten Makefiles zu generieren).
 3. Um Ihre C-Anwendung zu bauen, führen Sie einfach `build.bat` aus. Falls Sie Fehlermeldungen erhalten, stellen Sie sicher, dass die Pfade in `prologue.bat` und `CMakeLists.txt` (Variable `EASYLAB_DIR`) Ihrer Systemkonfiguration entsprechen.
Hinweis: Dies wird automatisch CMake neu ausführen, falls die Datei `CMakeLists.txt` verändert wurde.
 4. Hinweis: Das Bauen der Quelldateien wird im Verzeichnis `build` durchgeführt, das alle temporären Dateien enthält und bei Bedarf sicher gelöscht werden kann.

3.4 Uploading the firmware / Übertragen der Firmware

To upload a firmware image to the EasyKit Starter device, the following steps are needed:

Um die Firmware auf das EasyKit Starter zu übertragen, führen Sie folgende Schritte durch:

1. Open a command prompt (*Start* → *Run* → `cmd` → *OK*) and change to your workspace directory.
 2. Run the `upload.bat` batch file.
 3. When prompted, hit the EasyKit Starter's *Reset* button.
 4. While the device is in bootloader mode (indicated by LED5), hit any key on the keyboard to start the upload.
1. Öffnen Sie eine Eingabeaufforderung (*Start* → *Ausführen* → `cmd` → *OK*) und wechseln Sie in das Verzeichnis Ihrer Arbeitsumgebung.
 2. Führen Sie die Datei `upload.bat` aus.
 3. Wenn Sie dazu aufgefordert werden, betätigen Sie den *Reset*-Knopf am EasyKit Starter.
 4. Während sich Gerät im Bootloader-Modus ist (LED5 leuchtet), betätigen Sie eine beliebige Taste am Rechner, um die Übertragung zu starten.

4 Programming EasyKit Starter with *Assembler* / EasyKit Starter mit *Assembler* programmieren

Programming microcontrollers in *Assembler* is cumbersome and only advisable if one of the following situations apply:

- The application to develop is very small.
- The application or a part of it has to be executed very efficiently, which is why the programmer wants to have maximum influence on the instructions being generated (example: interrupt service routines).
- The microcontroller platform is very resource limited and there is not *C* compiler available.
- Usage for educational purposes.

Die Programmierung von Mikrocontrollern in *Assembler* ist mühsam und nur empfehlenswert, wenn einer der folgenden Punkte zutrifft:

- Die zu entwickelnde Anwendung ist sehr klein.
- Die Anwendung bzw. ein Teil davon soll hocheffizient ausgeführt werden, weshalb der Programmierer sehr genau Einfluss auf die auszuführenden Instruktionen nehmen will (Beispiel: Interrupt Service-Routine).
- Es steht nur eine sehr beschränkte Mikrocontrollerarchitektur zur Verfügung, für die kein *C*-Compiler existiert.
- Einsatz zu Ausbildungszwecken.

Since the complete programming of EasyKit Starter would be very complex (besides the actual program, one had to define startup routines for initializing the hardware and reservation of memory areas), this brief instruction is limited to so-called *inline Assembler*, i.e. *Assembler* code is embedded into *C* code. The advantage of this approach is that existing startup routines from EasyLab can be used and one may still write *Assembler* code.

The following web page gives an overview of the most important functions. However, it is not affiliated with EasyKit Starter in any way. Follow the instructions to write your own *Assembler* code:

Da die komplette Programmierung von EasyKit Starter mit *Assembler* sehr komplex wäre (neben dem eigentlichen Programm müssten beispielsweise noch Startup-Routinen für das Initialisieren der Hardware und das Anlegen von Speicherbereichen implementiert werden), beschränkt sich diese Kurzanleitung auf so genanntes *Inline Assembler*, d.h. *Assembler*-Code wird in *C*-Code eingebettet. Der Vorteil bei diesem Ansatz besteht darin, dass die existierenden Startup-Routinen aus EasyLab genutzt werden können und trotzdem *Assembler*-Code notiert werden kann.

Die folgende Webseite bietet einen Überblick über die wichtigsten Funktionen. Diese Webseite steht jedoch nicht mit EasyKit Starter in Verbindung. Folgen Sie den Anweisungen, um eigenen *Assembler*-Code zu schreiben:

<http://www.ethernut.de/en/documents/arm-inline-asm.html>