

## **Tutorial SW\_EMF01**

### **Setup CooCox**

Berner & Mattner Systemtechnik GmbH  
Erwin-von-Kreibitz-Straße 3  
80807 München

## **INHALTSVERZEICHNIS**

<b>1</b>	<b>LERNZIELE .....</b>	<b>2</b>
<b>2</b>	<b>BENÖTIGTE MATERIALEN .....</b>	<b>2</b>
<b>3</b>	<b>ERSTELLEN EINES NEUEN PROJEKTES .....</b>	<b>2</b>
<b>4</b>	<b>IMPORTIEREN EINES PROJEKTES UND FUNKTIONSDEFINITIONEN .....</b>	<b>4</b>
<b>5</b>	<b>PRIMÄRE FUNKTIONEN DES DEBUGMODUS .....</b>	<b>5</b>
<b>6</b>	<b>ÄNDERUNGEN AM PROGRAMM .....</b>	<b>6</b>

## 1 LERNZIELE

- Einrichten der Toolchain
- Kennenlernen von CoIDE
- Laden und Erstellen von Projekten
- Kennenlernen des Evaluation-Boards
- Kennenlernen der Debug-Möglichkeiten

## 2 BENÖTIGTE MATERIALEN

- STM32F4-Discovery Board
- USB-Kabel (A auf Mini-B)
- CoIDE (**Vers.: 1.7.5 !**)
- ARM-C-Compiler
- Projektdatei: *Tut00 - Setup.rar*
- (eventuell ST-Link USB Treiber → en.stsw-link009.zip)
- (evtl. STM32F4xx User Manual)

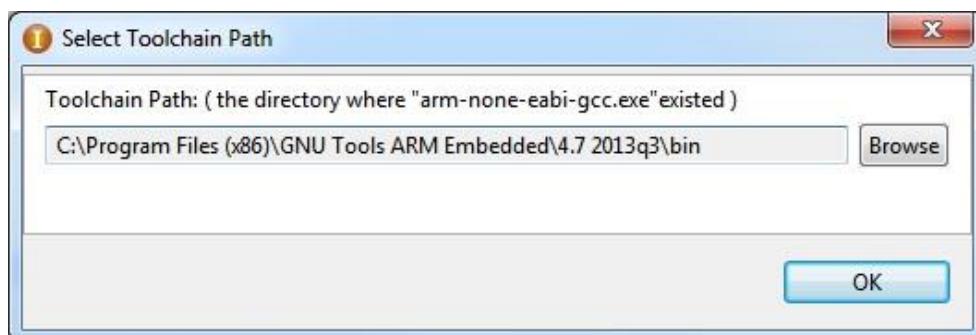
Alle benötigten Installer findest du [hier](#) zum Download (Achtung!\*).

Das benötigte Projekt findest du [hier](#) zum Download.

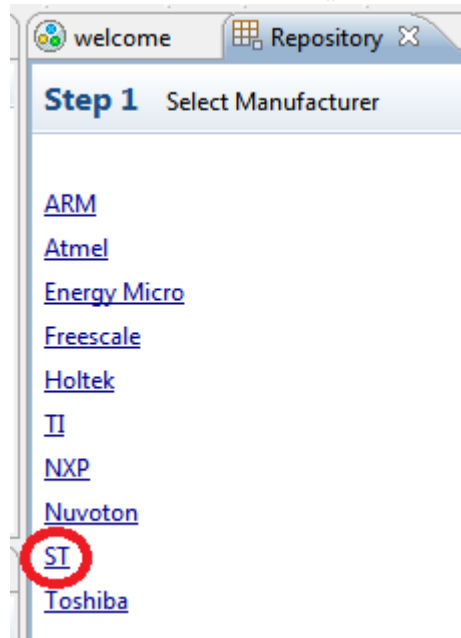
## 3 ERSTELLEN EINES NEUEN PROJEKTES

Die Toolchain wird eingerichtet. Ein neues Projekt stm32f4\_ex0a wird erstellt.

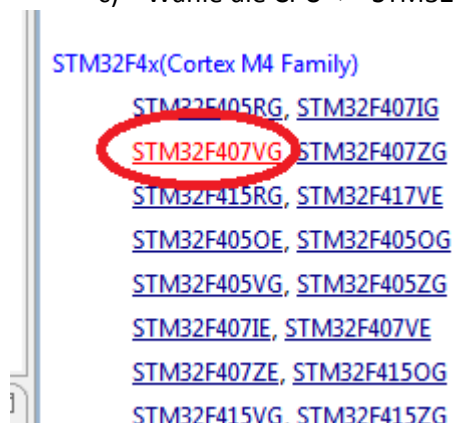
1. Installiere CoIDE. Folge dazu den Anweisungen des Installationswizards.
2. Installiere den GCC Compiler ARM-C-Compiler. Folge dazu den Anweisungen des Installationswizards und notiere den Installationspfad.
3. Starte CoIDE und gib im Menü unter "Projekt/Select toolchain Path" den Pfad der GCC-Installation an. Hier muss der Pfad zum "Bin-Verzeichnis" eingetragen werden also z.B. "C:\Program Files (x86)\GNU Tools ARM Embedded\4.7 2012q4\bin"



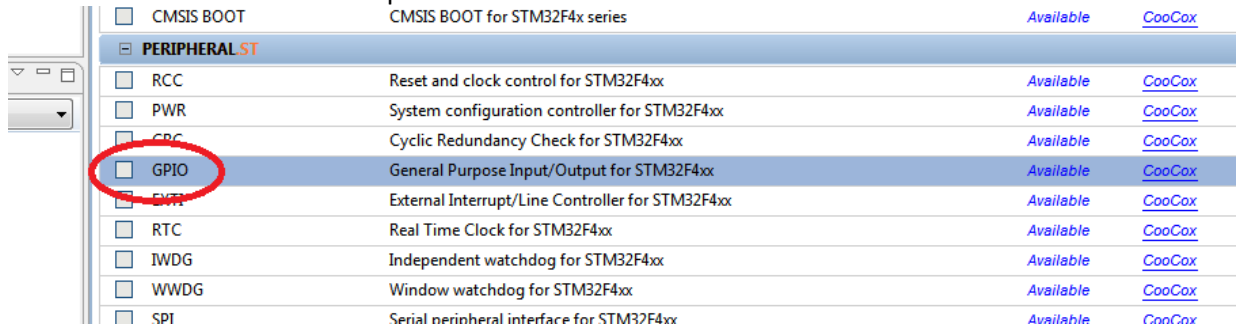
4. Lade und installiere den USB-Programmierer "en.stsw-link009.zip" für das STM32F4-Discovery. Diesen findest Du unter folgendem Pfad:  
/.../04\_Tutorials/03\_Installer/STM\_Board\_USB\_Driver
5. Verbinde das Eval-Board und den PC mit Hilfe des USB-Kabels. Beim ersten Anschluss ist es möglich, dass der Treiber nicht automatisch aktualisiert wird. In diesem Fall wird im Gerätemanager neben dem ST-Link/V2 ein gelbes Ausrufezeichen angezeigt. Um dies zu beheben klicke mit der rechten Maustaste auf den ST-Link/V2 und lasse den PC über „Treibersoftware aktualisieren“ automatisch nach aktueller Treibersoftware suchen.
6. Erstelle nun ein erstes Projekt.
  - a) Starte ColIDE und wähle auf der Startseite den Punkt „Browse im Repository“
  - b) Wähle den CPU Hersteller „ST“



- c) Wähle die CPU -> "STM32F407VG". Die findest du unter der „Cortex M4 Family“.



- d) Da wir unter anderem eine LED blinken lassen wollen, benötigen wir die Libraries für die GPIO (General Purpose Input/Output) Kontaktstifte. Wähle hierfür die Komponente GPIO unter Peripherals.



- e) Das Anklicken der Komponente löst einen neuen Dialog aus, der uns fragt, ob wir ein neues Projekt anlegen wollen. Klicke hier auf „Ja“ und speichere das Projekt im gewünschten Pfad.

Das Projektverzeichnis wird mit allen für das Eval-Board benötigten Libraries und Startup-Files erzeugt. Des Weiteren wird die *main.c* angelegt und das gesamte Projekt kompiliert.

#### Erlerner Inhalt:

- C-Projekt erstellen

## 4 IMPORTIEREN EINES PROJEKTES UND FUNKTIONSDEFINITIONEN

Ein Projekt wird importiert und der Code mit dem aus der 1. und 2. Aufgabe verglichen.

1. Importiere das Projekt „stm32f4\_ex0\_setup“ wie folgt in den Workspace:
  - a. Project → Open Project
  - b. Öffne die Datei stm32f4\_ex0\_setup.coproj
2. Betrachte nun die *main.c*. Diese Datei beginnt mit dem include-Statements für die Header-Datei *stm32f4xx\_gpio.h*. Öffne diese durch Halten der [STRG]-Taste und Linksklick mit der Maus auf den include-Befehl im Programmcode.

Die Datei *stm32f4xx\_gpio.h* enthält die Firmware für die GPIO Peripherie und liegt, da wir sie für dieses Projekt ebenfalls im Repository ausgewählt wurde, im Verzeichnis Projektordner\cmsis\_lib\includeb von wo sie durch das #include Statement in unser Programm eingebunden wird.

3. In der *main.c* überzeuge dich, dass du das erzeugte Beispielprogramm verstehst. Verfolge dazu die Definitionen der aufgerufenen Funktionen durch Halten von [STRG] und Linksklick der Maus auf die Funktionsaufrufe oder auf die Bezeichnung der LEDs z.B. LED3. Wie zuvor gelangt man zu der Stelle im Programmcode an der diese definiert wurden. Mache dich mit dem Programm vertraut und überlege dir was das Programm bewirkt.

**Erlerner Inhalt:**

- C-Projekt importieren
- Funktionsdefinition aufrufen

## 5 PRIMÄRE FUNKTIONEN DES DEBUGMODUS

Die Funktionen des Debuggers werden genauer betrachtet.

1. Klicke auf Project → Build (F7) oder den Build Button in der Toolbar um den Code zu kompilieren. Das Programm ist hierdurch noch nicht gestartet.
2. Mit dem Befehl Debug → Debug (CTRL + F5) oder den Debug-Button in der Toolbar startet man die Ausführung des Programms im Debugmodus.

ColIDE flasht jetzt den Code auf das Board und versucht eine Verbindung zum ST-Link Debugger aufzubauen, um die Programmausführung im Debugmodus verfolgen zu können. Dazu muss das Eval-Board mit dem PC verbunden sein. Ist die Verbindung erfolgreich hergestellt worden öffnet sich die Debugansicht.

Falls die Verbindung nicht hergestellt werden konnte, ist entweder das Board nicht korrekt angeschlossen.

Durch erneutes Starten des ‚Debug-Modus‘ kann nun das Programm gestartet und die Ausführung in der Debug-Ansicht verfolgt werden.

3. Klicke auf *Run* oder F5 um das Programm zu starten und den Step-Modus zu verlassen. Mit *Suspend* (F9) kann der Ablauf wieder unterbrochen werden.
4. Benutze nicht die Funktion *Run* sondern *Step Into* (F11) oder *Step Over* (F10)
  - *Step Over*: Nur die angewählte Code Zeile wird ausgeführt
  - *Step Into*: Die angewählte Code Zeile wird ausgeführt und es wird in die Funktionsdefinition der auszuführenden Zeile gesprungen (wie bei [STRG]-Linksklick)
5. Klicke dich mit *Step Into* (F11) und oder *Step Over* (F10) durch das Gesamte Programm. Verdeutliche dir was in den einzelnen Programmzeilen passiert
6. Klicke in der Debugansicht auf *Terminate* oder Drücke [CTRL] + F5 um den Debugmodus zu verlassen. Die Ansicht wechselt zurück in die C/C++ Perspektive.
7. Das Board wird resettet und das Programm läuft auf dem Board weiter bis dieses von der Spannungsversorgung getrennt wird oder der Flashspeicher mit einem neuen Programm beschrieben wird. Wenn die Durchführung der Schritte problemlos funktioniert hat, ist die Toolchain zum Entwickeln auf dem STM32F4-Discovery Evaluation-Board nun fertig eingerichtet.

**Erlerner Inhalt:**

- Primäre Funktionen des Debugmodus
- Debugmodus starten und beenden

## 6 ÄNDERUNGEN AM PROGRAMM

Zum Abschluss wird das Programm aus `stm32f4_ex0_setup` variiert und die LEDs werden in einer Endlosschleife nacheinander angehen und nacheinander ausgehen. Gehe dazu wie folgt vor:

1. Kopiere die Befehle zum Setzen und Löschen der Bits zum Ein- und Ausschalten der LEDs in die Endlos-While-Schleife.
2. Erweitere die leere Funktion `void delay(uint32_t time)` am Ende der main-Funktion um eine while Schleife zum Runterzählen einer Zählvariable. Ersetze mit dieser Funktion die vorherigen while-Zählschleifen.
3. Bringe das Programm zum Laufen.

### **Erlerner Inhalt:**

- Erste Programmänderungen

Das war das erste Tutorial zur Inbetriebnahme und zum Kennenlernen des Evaluationboards und der Toolchain.