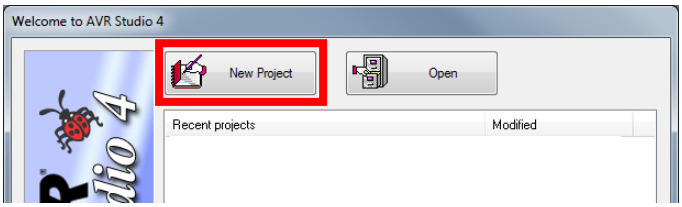
LABORÜBUNGEN MIKROCONTROLLER – ÜBUNG 1

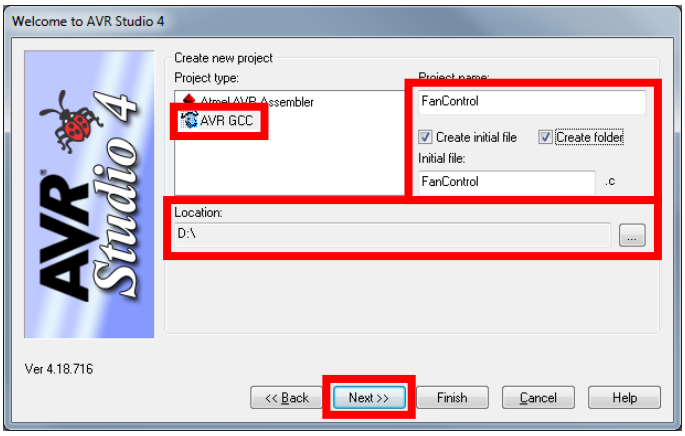
BEISPIEL 1 – PROJEKT ANLEGEN

Starten Sie das Programm AVR Studio 4 und erstellen Sie anhand der nachfolgend zur Verfügung gestellten Anweisung ein Beispielprojekt. Alternativ können Sie aus dem MIC-LB-Kurs im Moodle ein fertiges Grundgerüst herunterladen und die *.aps* Projektdatei öffnen.

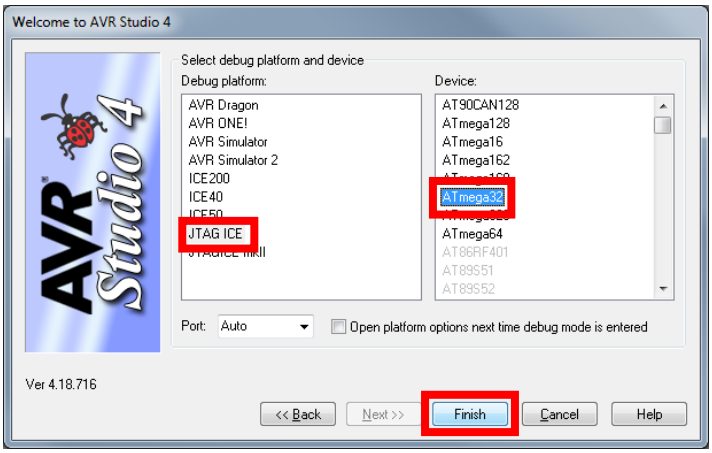
Erstellen Sie mit dem Button *New Project* ein neues Projekt. Falls der Wizard beim Start von AVR Studio nicht zur Verfügung steht, finden Sie diesen unter dem Menüpunkt *Project* → *Project Wizard*.



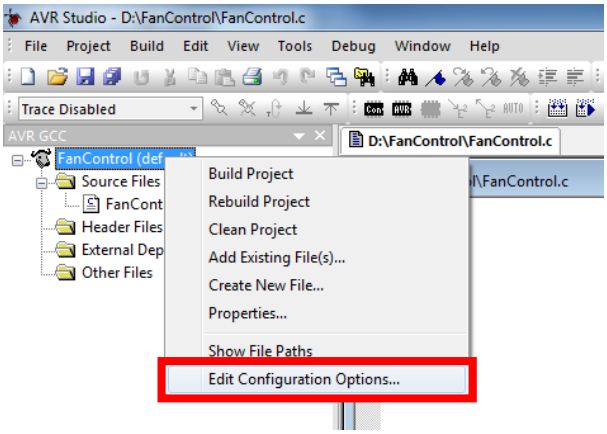
Im nächsten Schritt als Projekttyp *AVR GCC* selektieren. Als Projektname *FanControl* vergeben und verifizieren, dass beide Checkboxen *Create initial file* und *Create folder* gesetzt sind. Unter Location einen geeigneten Pfad vergeben und mit *Next* bestätigen.



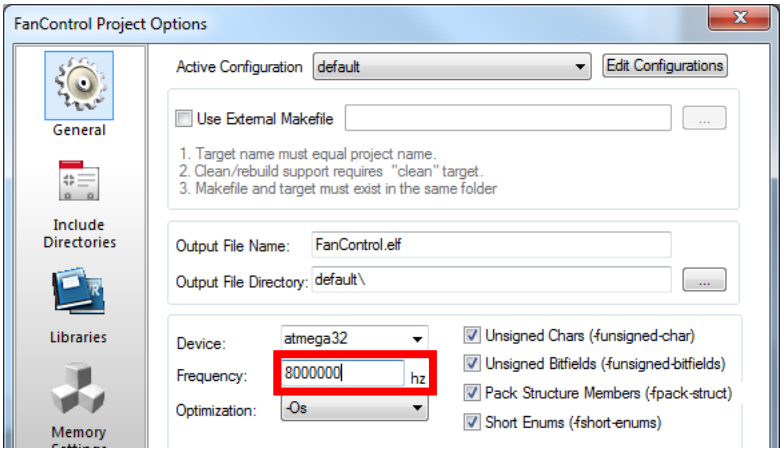
Im abschließenden Dialog als *Debug platform* den verwendeten *JTAG ICE* selektieren und als *Device* den Mikrocontroller *ATmega32* auswählen. Mit *Finish* bestätigen.



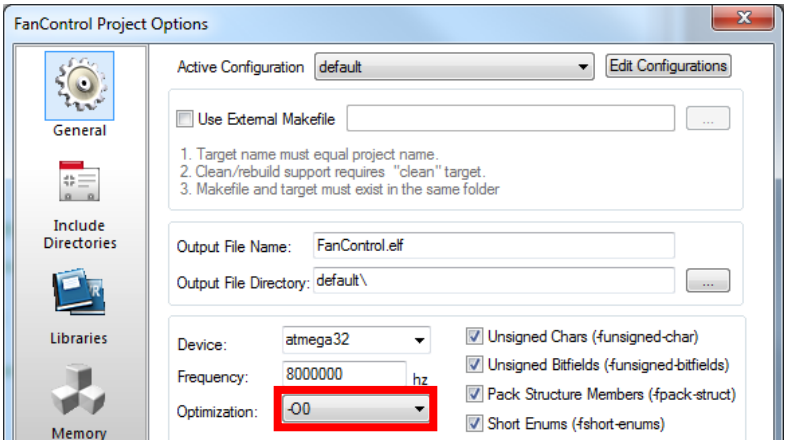
Klicken Sie im AVR GCC Dateiexplorer mit einem Rechtsklick auf den Wurzelknoten und wählen Sie den Menüpunkt *Edit Configuration Options…* aus.



Definieren Sie im Dialogfenster für die Projekteinstellungen unter dem Punkt *Frequency* die Oszillatorfrequenz von 8MHz. und bestätigen diese Einstellung mit *OK*.



Der Optimierungsgrad des Compilers wird von *-Os* (Optimieren auf Codegröße) auf *-O0* (Keine Optimierung) abgeändert, damit beim Debuggen Einzelschritte besser nachvollziehbar sind.



Bestätigen Sie die abgeänderte Konfiguration mit *OK*.

Öffnen Sie die Datei *FanControl.c*, welche im AVR GCC Dateiexplorer zu finden ist. Tragen Sie folgendes Grundgerüst in die Datei ein:

```
/**
* @mainpage Fan Control MIC-LB WS2011
*
* Fan Control ist ein Semesterprojekt im Rahmen des Mikrocontrollerlabors,
* welches im Zuge des 5. Semesters am Studiengang Informationstechnik & System-Management
* an der Fachhochschule Salzburg durchgeführt wird.
*
* @author Jane Doe
* @author John Doe
*
* @date 12.09.2011
*/

/**
* @file FanControl.c
* @brief Kurze Beschreibung
*
* Lange Beschreibung
*/

/**
* @brief Beispieldokumentation einer Funktion in Doxygen
*
* Dieser Funktionsprototyp dient alleinig zur Veranschaulichung der Doxygen-Dokumenation einer
* Funktion.
*
* @param[out] out_szOut Ausgabestring
* @param[in] in_szIn Eingabestring
*
* @return int Gibt den Status der Operation zurück
*
* @todo Aus dem Projekt entfernen.
*
* @bug Implementierung fehlt
*/
int foo(char *out_szOut, char *in_szIn);

/**
* @brief Haupteinsprungspunkt
*
* Hier soll in Folge die Initialisierung der einzelnen Peripherie
* sowie deren Scheduling durchgeführt werden.
* Eine Endlosschleife dient dabei als einfacher Scheduler.
*
* @return int Return-Code für die korrekte oder fehlerhafte Ausführung des Hauptprogramms
*
* @todo Implementieren
*/
int main(void)
{
        while(1)
        {

        }
        return 0;
}
```

Dieses Grundgerüst beinhaltet bereits Kommentare nach der Doxygen-Konvention. Doxygen wird in Folge zur automatischen Quellcode-Dokumentation herangezogen.

Der erste Kommentarblock dient alleinig zur Gesamtbeschreibung des Projekts. Hierbei sollte von Ihnen das Feld des Autors angepasst sowie Ihr Projektpartner eingetragen werden.

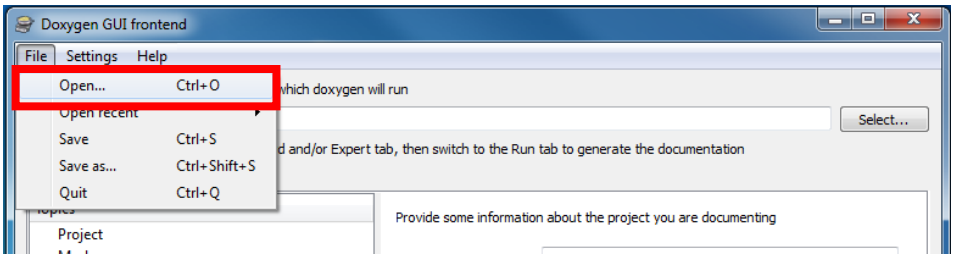
Der zweite Kommentarblock kann für jede Datei als Vorlage verwendet werden.

Der restlichen Kommentarblöcke dienen der direkten Beschreibung der einzelnen Funktionen. Die Funktionsdeklaration *foo* dient dabei zur Veranschaulichung der Doxygen-Syntax. Die Deklaration kann gemeinsam mit der Dokumentation in Folge aus dem Projekt entfernt werden.

BEISPIEL 2 – DOXYGEN DOKUMENTATION

Laden Sie die Datei *Doxygen-Vorlage* aus dem MIC-LB-Kurs im Moodle lokal in das Projektverzeichnis. Die Datei muss sich auf gleicher Ebene wie die Quelldateien und der Projektdatei befinden.

Starten Sie das Programm *Doxywizard* und laden Sie die Vorlage unter *File* → *Open*. Navigieren Sie zum Projektverzeichnis, in dem Sie die Vorlage zuvor gespeichert haben.



Unter Step 1 sollte nun das Projektverzeichnis konfiguriert sein. Gehen Sie auf den Reiter *Run* und führen Sie die Erstellung der Dokumentation mit dem Button *Run doxygen* aus. Nach Abschluss der Kompilierung können Sie mit dem Button *Show HTML output* die Dokumentation als HTML-Datei betrachten.

BEISPIEL 3 – PORT TOGGELN

Legen Sie im AVR GCC Dateiexplorer eine neue Quelldatei mit dem Namen *PortToggle.c* sowie eine dazugehörige Headerdatei mit dem Namen *PortToggle.h* an.

Dokumentieren Sie beide Dateien entsprechend der Doxygen-Konvention aus dem Grundgerüst.

Deklarieren Sie alle verwendeten Funktionen innerhalb der Quelldatei PortToggle.c in der dazugehörigen Headerdatei und binden Sie letztere in das Hauptprogramm FanControl.c ein.

Stellen Sie folgende Funktionen bereit:

- InitPortToggle
- TogglePort

Die InitPortToggle-Funktion wird im Hauptprogramm vor der Endlosschleife aufgerufen und initialisiert den Port.

Die TogglePort-Funktion wird innerhalb der Endlosschleife ausgeführt und soll den Status von Port A im Halb-Sekundentakt invertieren. Die Verzögerung soll dabei mit einer blockierenden Funktion erfolgen.

Binden Sie dafür folgende Headerdatei in PortToggle.c ein und verwenden Sie deren Funktionen:

```
#include <util/delay.h>
```