Praktikum Mikrocontrollertechnik SS 2014
Versuch 1

Stand: 5. 5. 2014

Einführung in die C-Programmierung Atmel AVR. Die Programmiersprache C ist alt und von Grund auf häßlich. Es ist aber die am meisten verwendete Programmiersprache.

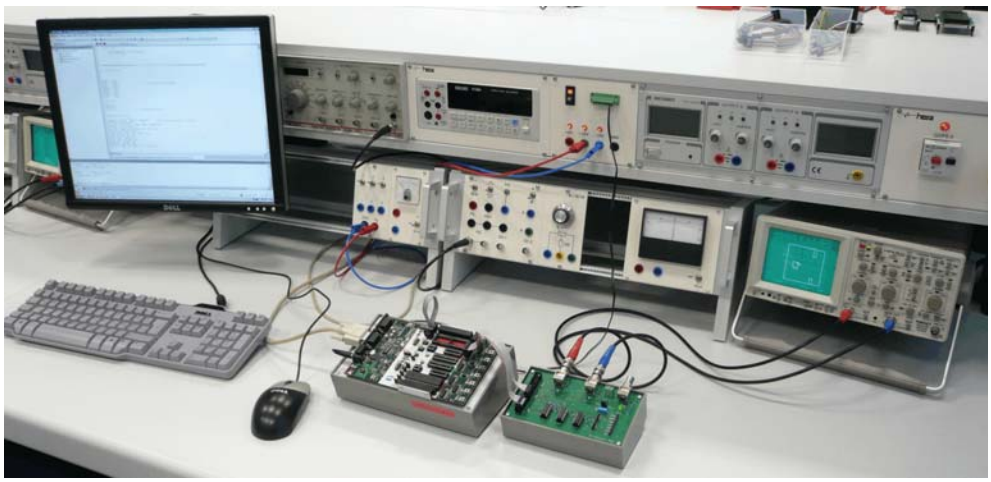
Aufgaben:

1. Kennenlernen des grundsätzlichen Entwicklungsgangs.
2. Darstellen von Zeit mittels Software.
3. Elementare Anwendungsprogrammierung.

Ausrüstung:

- AVR Studio 4 (4.19),
- WinAVR C-Compiler (AVR GCC),
- Starterkit AVR STK 500,
- Universaladapter 10b (die Bunte Kuh),
- XY-Adapter 09,
- Joystick 12,
- Siebensegmentanzeige 09/13.

Zu den Geräten siehe die jeweilige Kurzbeschreibung, zum Starterkit auch die Originaldokumentation der Fa. Atmel.



Aufgabe 1: Den Entwicklungsgang kennenlernen

Jedes Vorhaben ist ein Projekt. Wenn Sie etwas Neues beginnen wollen, müssen Sie im AVR Studio ein neues Projekt einrichten.

1. Software starten.
Die Anwendung heißt AVR Studio 4.

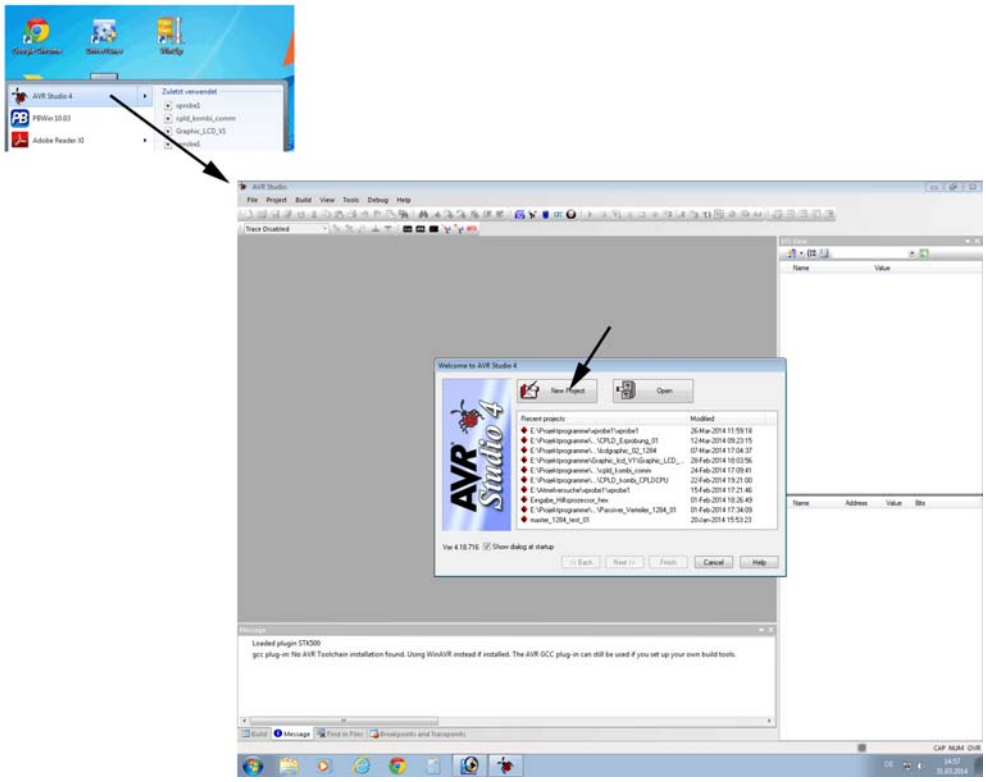


Abb. 1 Das AVR Studio wurde gestartet. Wir wollen eine neues Projekt einrichten.

2. Ein neues Projekt anlegen.

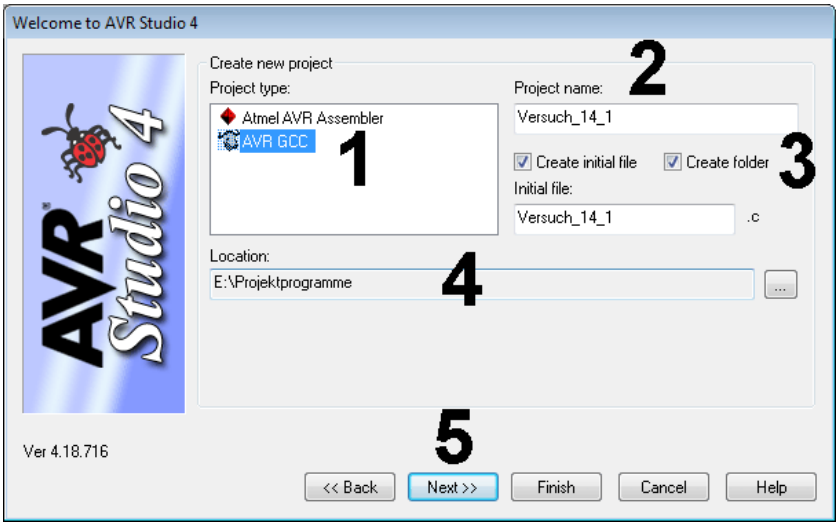


Abb. 2 Das Projekt benennen und anlegen. 1 - es soll in C geschrieben werden. 2 - Namen eintragen. 3 - diese Kontrollkästchen müssen beide aktiv sein. 4 - bitte ein vernünftiges Verzeichnis auswählen. Das Projekt nicht auf dem Desktop oder auf der Netzfestplatte ablegen... 5 - weiter.

3. Den Mikrocontroller auswählen.

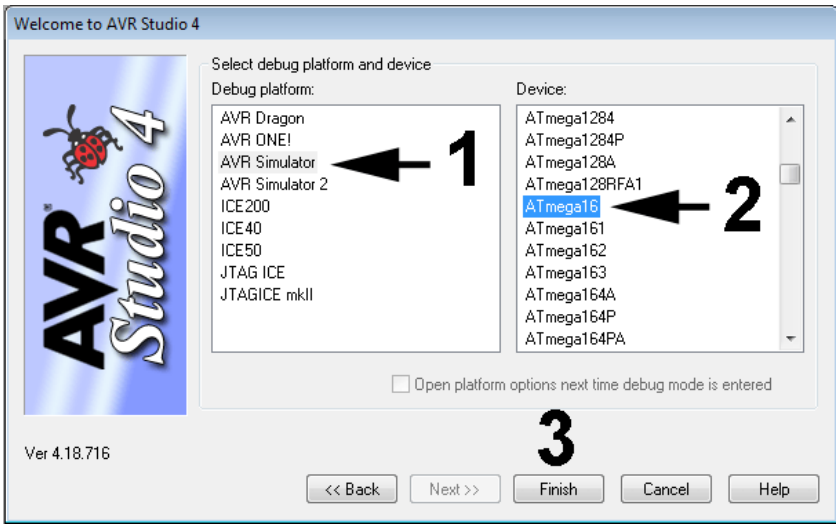


Abb. 3 Fehlersuchplattform und Mikrocontroller auswählen. 1 - wir wählen den AVR Simulator zum Verfolgen der Programmabläufe (obwohl wir ihn nicht brauchen und unsere Fehler zu Fuß suchen wollen). 2 - unser Mikrocontroller ist der Atmega 16. Er ist leistungsfähig genug und ausreichend ausgestattet, aber nicht zu kompliziert. 3 - Projekterstellung beenden.

4. Ein Musterprogramm übernehmen und übersetzen.

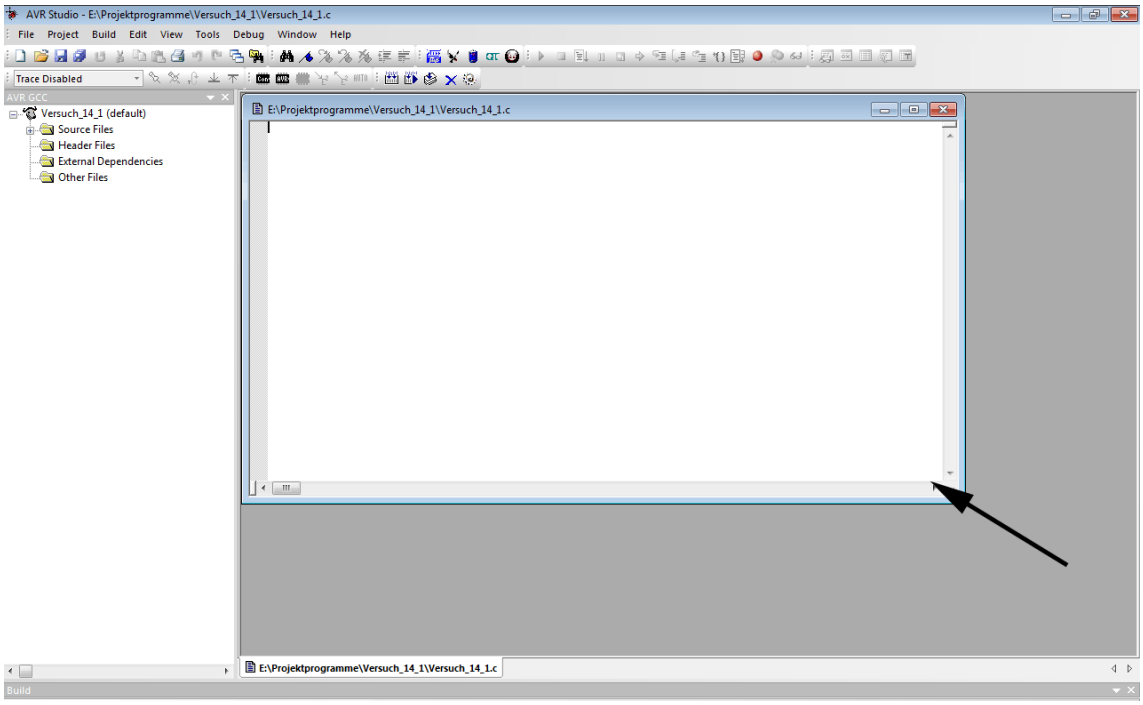


Abb. 4 Das Projekt ist geöffnet. Der Pfeil zeigt auf das (leere) Eingabefenster des Programmeditors.

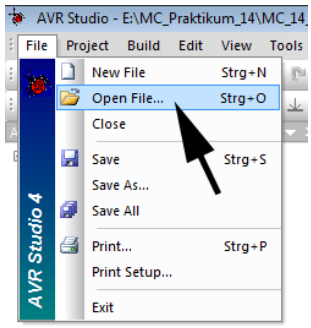


Abb. 5 Wir wollen ein Musterprogramm übernehmen.

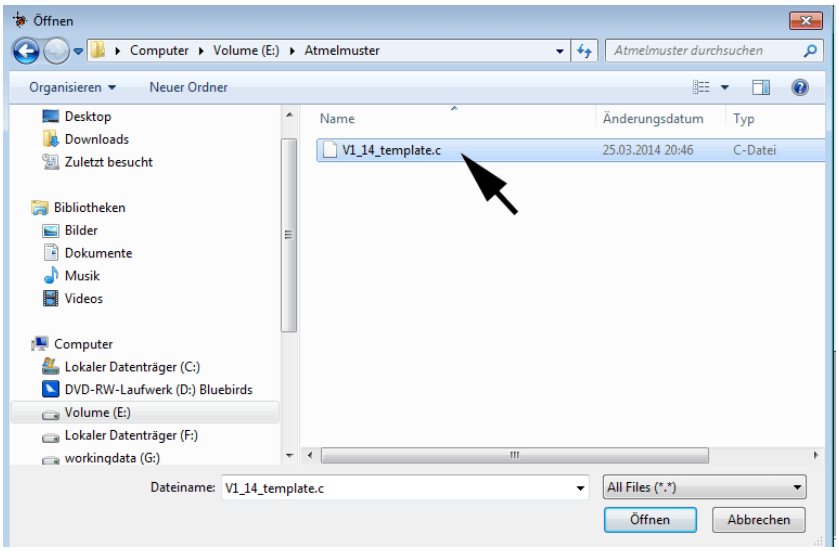


Abb. 6 Das Musterprogramm auswählen und öffnen. Es heißt "V1_14_template.c".

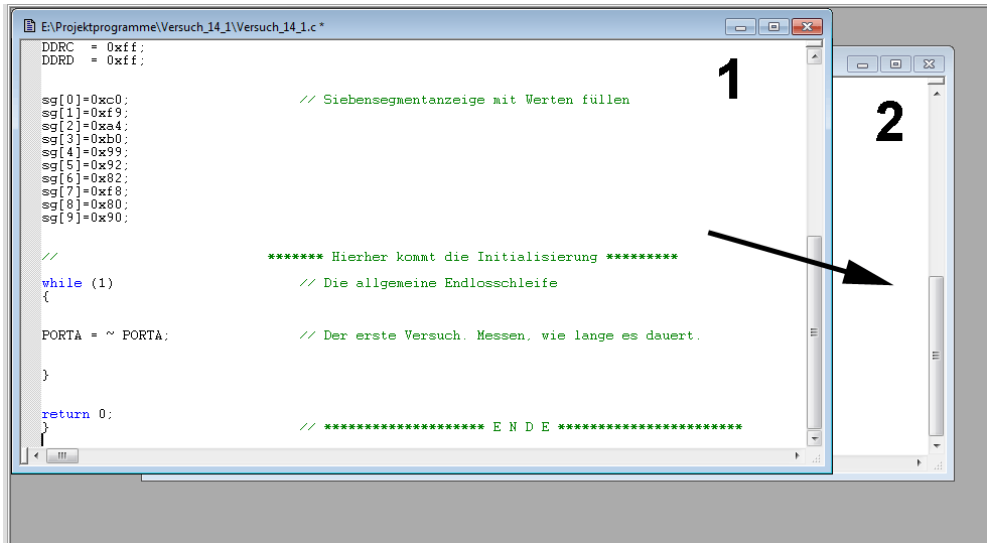


Abb. 7 Das Musterprogramm wird ins Projekt übernommen. 1 - Musterprogramm; 2 - Projektprogramm. In 1 alles markieren. Kopieren. in 2 einfügen. 1 schließen.

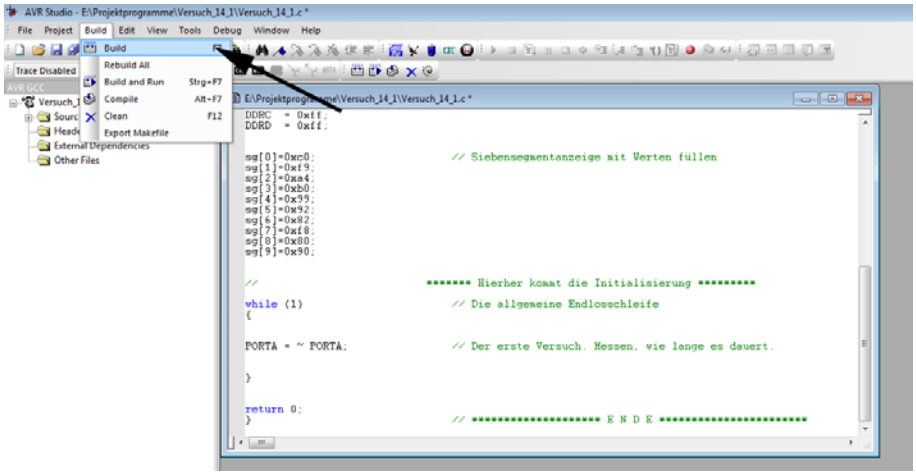


Abb. 8 Jetzt wird das Programm übersetzt. Auswahl “Build” – “Build”.

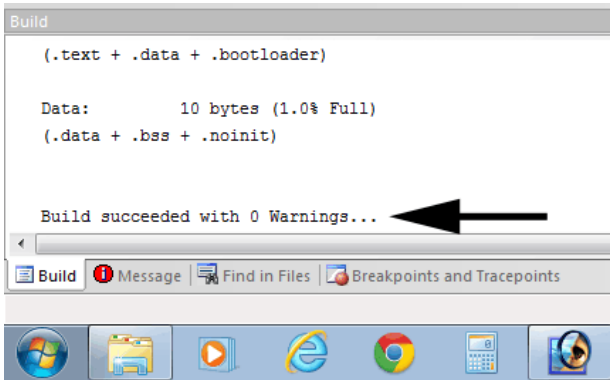


Abb. 9 Diese Nachricht (Pfeil) ist entscheidend. Jetzt kann der Mikrocontroller programmiert werden.

5. Den Mikrocontroller programmieren.

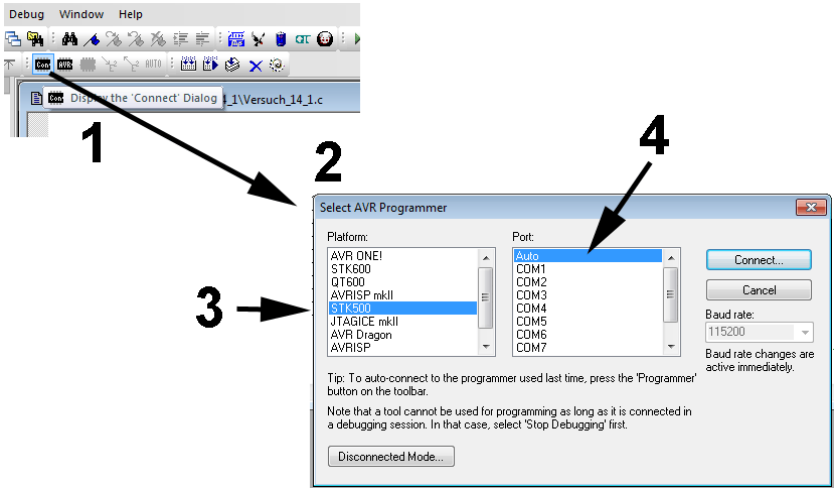


Abb. 10 Der Programmer wird ausgewählt und aktiviert. 1 - mit dem Programmer verbinden. 2 - der Auswahldialog. 3 - es ist das Starterkit STK 500. Es ist an eine serielle Schnittstelle angeschlossen. 4 - wir lassen das Programm die Schnittstelle selbst finden.

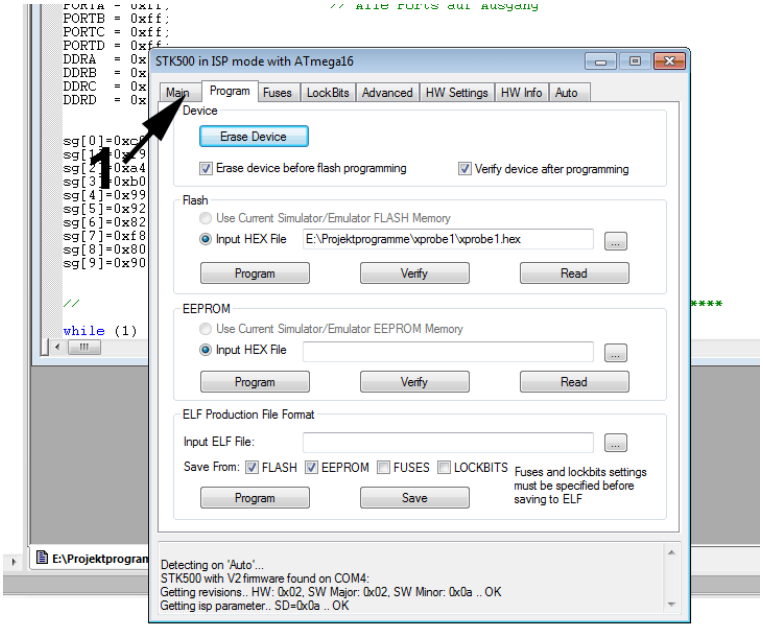


Abb. 11 So meldet sich der Programmer. 1 - wir sollten aber nicht gleich programmieren, sondern erst einmal hier nachschauen, ob auch alles stimmt.

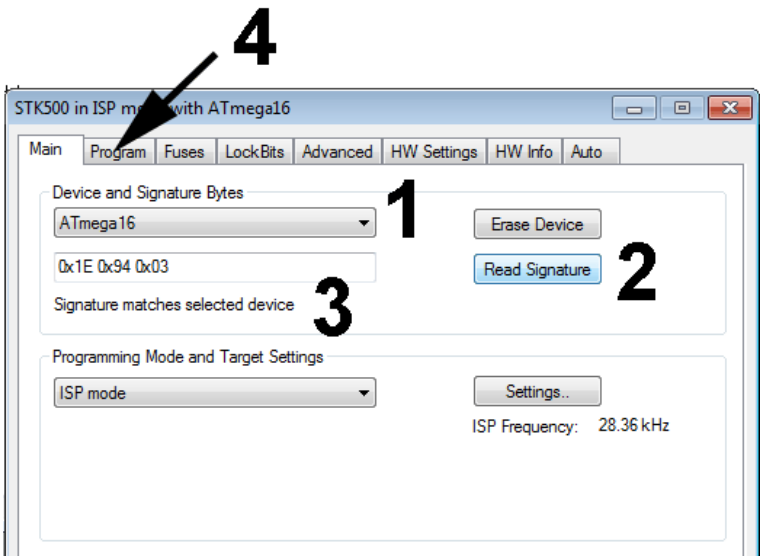


Abb. 12 Vertrauen ist gut, Kontrolle ist besser... 1 - ist der richtige Controllertyp ausgewählt? 2 - zur Kontrolle die Signatur auslesen. 3 - diese Aussage ist entscheidend. Der Schaltkreis auf dem Starterkit ist tatsächlich vom ausgewählten Typ. 4 - zurück zum Programmieren.

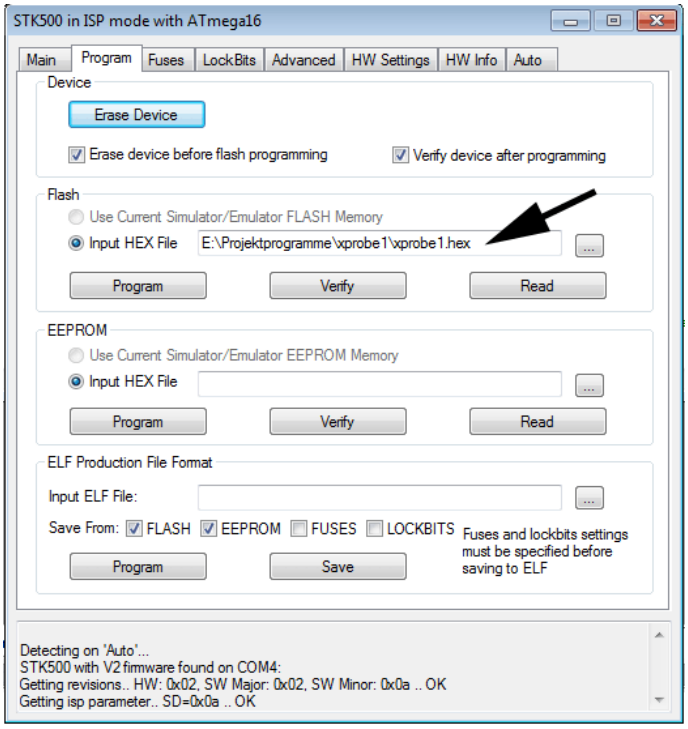


Abb. 13 Das Programm muß in den Flash. Hierzu müssen wir zunächst die Programmdatei auswählen. Der Programmer hat keinen automatischen Datenverbund mit der Programmentwicklung.

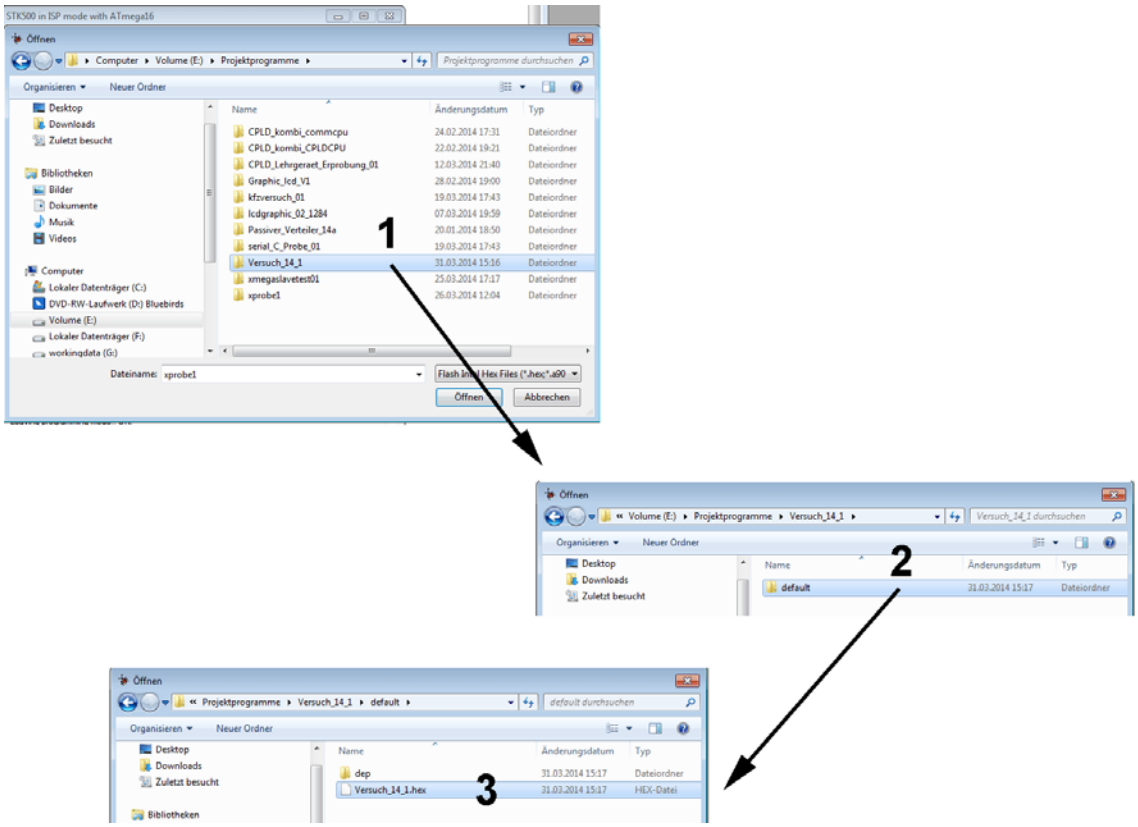


Abb. 14 Programmierdateiauswahl. 1 - das Verzeichnis des Projekts öffnen. 2 - das Unterverzeichns “default” öffnen. 3 - wir brauchen die Datei mit der Endung .hex. Auswählen.

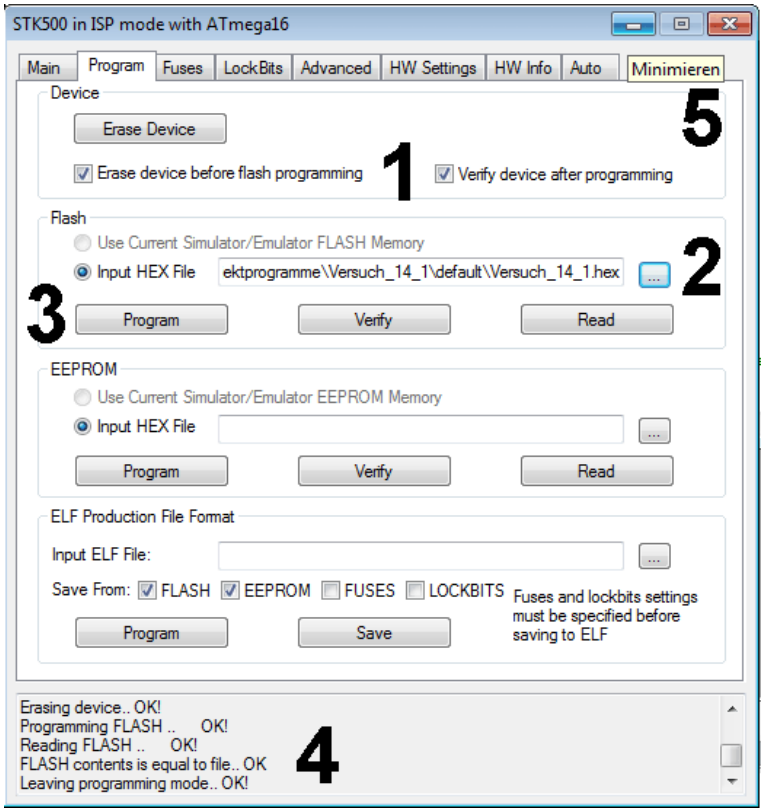


Abb. 15 Programmieren. 1 - diese beiden Kontrollkästchen müssen aktiv sein. 2 - die Programmierdatei ist ausgewählt. 3 -Programmieren auslösen. 4 - die Erfolgsmeldungen. Fertig. 5 - Programmierdialog nicht schließen, sondern minimieren (um ihn später wieder aufrufen zu können).

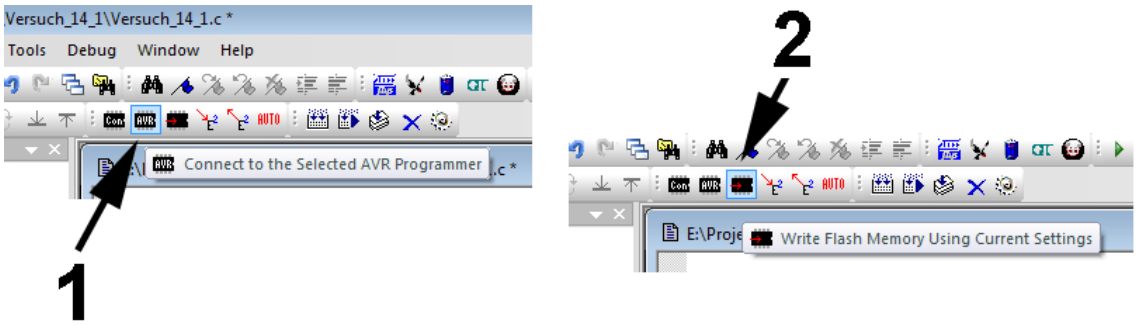


Abb. 16 Programmerauswahl. 1 - mit dem ausgewählten Programmer verbinden. 2 - den Flash gemäß den aktuellen Einstellungen neu programmieren.

Aufgabe 2: Den Aufbau eines C-Programms verstehen.

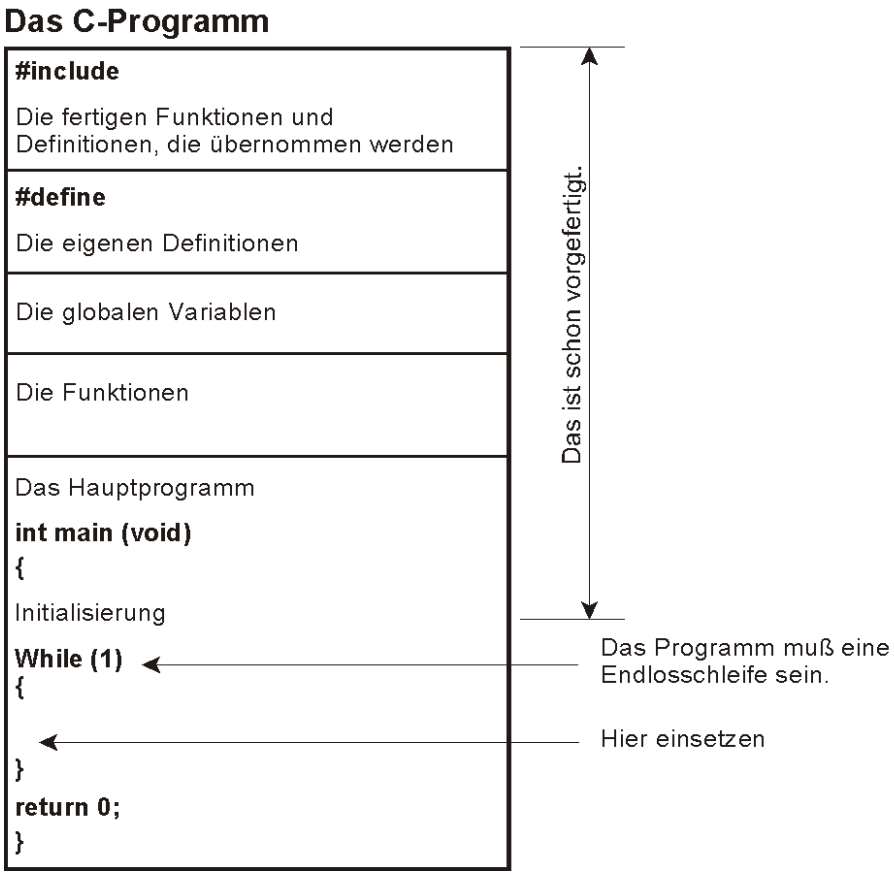


Abb. 17 Die grundsätzliche Struktur eines C-Programms.

Aufgabe 3: Programmlaufzeiten messen.

Eine beliebige Bitposition von Port A mit dem Oszilloskop verbinden (Tastkopf). Wir lassen das Musterprogramm laufen. In der Endlosschleife wird der Port A zyklisch umgeschaltet. Es müssen symmetrische Rechteckimpulse angezeigt werden. Damit kann die Durchlaufzeit der Schleife gemessen werden. Eine Halbperiode = ein Schleifendurchlauf. Zeit messen.

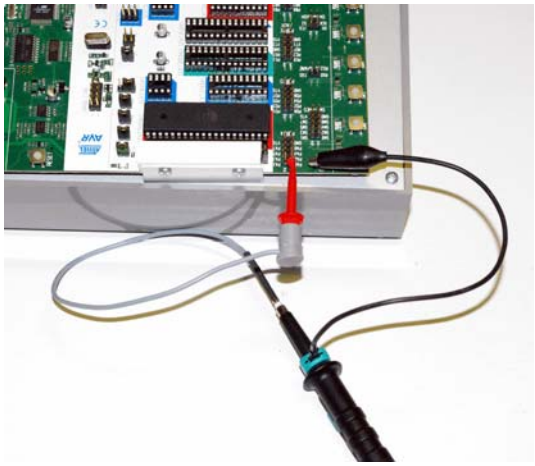


Abb. 18 Zeitmessung mit Oszilloskop.

Die Durchlaufzeit hängt vom Compiler ab. Die Konfigurationseinstellungen des Projekts aufrufen.

Welchen Maschinencode hat der Compiler erzeugt? Das ist aus dem sog. List File ersichtlich (Dateiendung .lss).

Der Maschinencode hängt von der gewählten Optimierungsstufe ab. O0 = keine Optimierung. Os = höchste Optimierung. Mit beiden Einstellungen probieren. List Files ansehen.

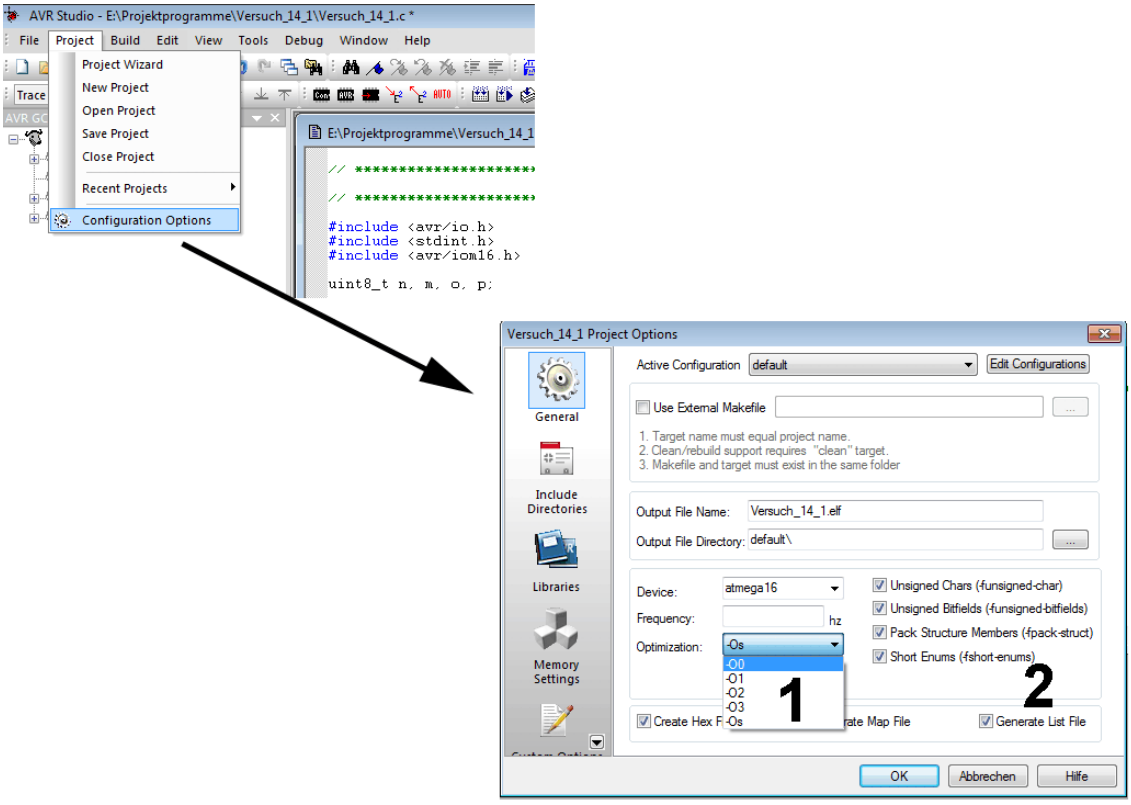


Abb. 19 Die Konfigurationseinstellungen des Projekts. 1 - Optimierung; 2 - List File erzeugen / nicht erzeugen.

Selbst programmieren

Um die folgenden Aufgaben zu lösen, müssen Sie selbst programmieren. Die Programmabläufe fügen Sie in mit dem Editor des AVR Studio in die geöffnete C-Quelldatei ein.

Achtung:
Der Compiler achtet auf die Groß- und Kleinschreibung (anders als der Assembler). Die Atmel-AVR-Definitionen (Ports, Steuerbits usw.) werden stets groß geschrieben, reservierte Bezeichner der Programmiersprache hingegen klein. Korrekt geschriebene reservierte Bezeichner werden blau wiedergegeben, Kommentare grün. Kommentarzeichen: // (zeilenweise) oder /* */ (ganze Blöcke). Die Semikolons nach den Anweisungen nicht vergessen!

Richtig	Falsch
while (1)	WHILE (1)
PORTA	porta
RXEN	Rxen

Programmbeispiele

Programmbeispiele finden Sie auf der C-Quelldatei “V1_14.examples.c”. Sie steht im selben Verzeichnis wie das Musterprogramm. Sie können diese Datei im Editor öffnen, und Programmstücke in die C-Quelldatei Ihres Projekts übernehmen. Manchmal sind zudem noch Änderungen erforderlich, beispielsweise in der Initialisierung der E-A-Ports. Es versteht sich von selbst, daß Sie auch alternative eigene Lösungen ausprobieren dürfen.

Aufgabe 4: Zeit darstellen.

Wir stellen die Projektkonfiguration auf O0 = keine Optimierung. Das Ziel ist eine Funktion millisecs(), der ein 16-Bit-Wert als Parameter übergeben wird. Die Funktion wartet so viele Millisekunden ab, wie dieser Wert angibt (Warteschleife).

Hierzu schreiben wir zunächst eine Funktion milli(), die genau eine Millisekunde wartet. Erprobung: durch Einfügen in die Endlosschleife. Die Durchlaufanzahl (eine Konstante) in der Funktion so lange ändern, bis der Schleifendurchlauf näherungsweise eine Millisekunde dauert.

Mit dieser Funktion die Funktion millisecs() schreiben und erproben. millisecs (n) ist einfach ein n-maliges Aufrufen von milli().

Wenn alles klappt, ausprobieren, was sich ergibt, wenn die Optimierung auf Volldampf (Os) geschaltet wird. Dann aber wieder zurück zu O0 (jetzt wissen wir nämlich, warum...).

Programmbeispiel 01.

Aufgabe 5: Lauflicht in eine Richtung.

Die LEDs des Starterkits mit Port C verbinden. Zyklische aufeinanderfolgende Erregung in eine Richtung. Die LEDs sind aktiv Low.

Abb. 20 Die LEDs an Port C.

Programmbeispiel: 02.

Aufgabe 6: Lauflicht in beide Richtungen (hin und her).

Zyklische aufeinanderfolgende Erregung der LEDs, zunächst in die eine Richtung, dann zurück.

Programmbeispiel: 03.

Aufgabe 7: Gebäudesystemtechnik (1).

Das elektrische Licht ein- und ausschalten. Tasten an Port D anschließen. Port D auf Eingang. Mit einer der Tasten (z. B. Taste 7) soll die gesamte LED-Reihe ein- und ausgeschaltet werden (erste Betätigung: ein, zweite Betätigung: aus usw.). Die Tasten sind aktiv Low.

Abb. 21 LEDs an Port C, Tasten an Port D.

Programmbeispiel: 04.

Aufgabe 8: Gebäudesystemtechnik (2).

Das Ein- und Ausschalten funktioniert nicht vollkommen sicher. Mal klappt's, mal nicht. Das liegt daran, daß die Taste prellt. Jetzt Entprellung (durch Zeitverzögerung) einbauen.

Programmbeispiel: 05.

Aufgabe 9: Gebäudesystemtechnik (3).

Das elektrische Licht nacheinander ein- und ausschalten (Power Sequencer). Am Anfang sind alle LEDs dunkel. Mit einer der Tasten eine Einschaltsequenz auslösen. Zunächst schaltet nur die erste LED, nach etwa 300... 500 ms kommt die zweite hinzu usw., bis schließlich alle acht eingeschaltet haben. Wird dieselbe Taste erneut betätigt, sollen die LEDs in umgekehrter Reihenfolge nacheinander ausschalten.

Programmbeispiel: 06.

Aufgabe 10: XY-Darstellung.

Den XY-Adapter 09 an die Ports B und C anschließen und mit dem Oszilloskop verbinden. Oszilloskop auf XY-Betrieb schalten.

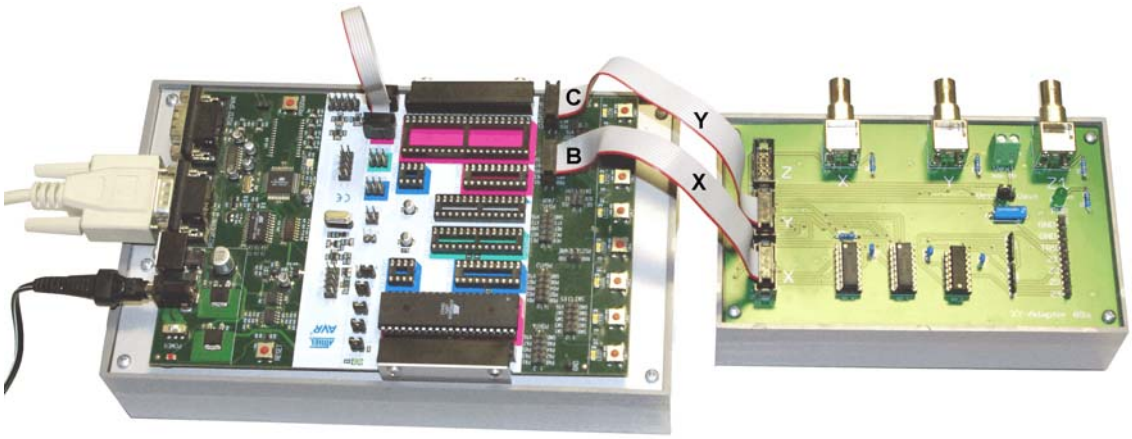


Abb. 22 Starterkit mit XY-Adapter 09. Verbinden: Port C mit Y, Port B mit X.

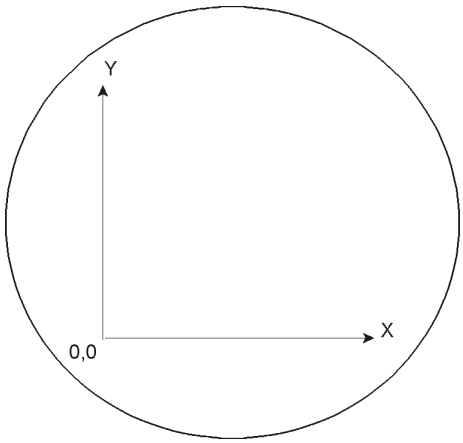


Abb. 23 Das Prinzip der Darstellung auf dem Oszilloskop.

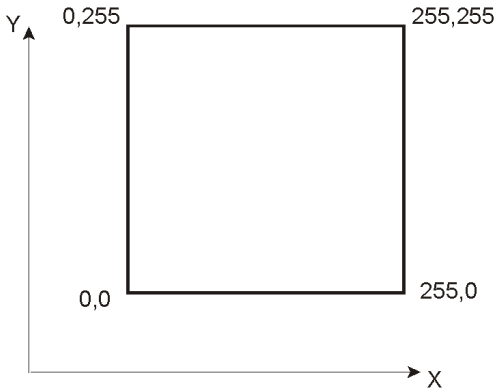


Abb. 24 Die Grenzen des Darstellbereichs. Die Darstellung besteht aus Bildpunkten (Pixeln). In jeder Koordinatenachse sind es maximal 256 Pixel.

1. Eine waagrechte Linie maximaler Länge.
Zunächst naiv mit einer FOR-Schleife probieren.

Programmbeispiel: 07.

2. Eine senkrechte Linie maxinaler Länge anhängen.
Das klappt offensichtlich nicht. Weshalb?

Programmbeispiel: 08.

3. So umbauen, daß es klappt.
Es ist aber immer noch ein Pixel zuwenig. Wie kann man das korrigieren? (Ob es zuwenig Pixel sind oder nicht, können wir sehen, wenn wir zusätzlich Pixel in die äußersten Ecken schreiben und sie etwas länger stehenlassen..)

Programmbeispiel: 09.

4. Ein Quadrat maximaler Größe.
Die fehlenden Pixel werden zusätzlich eingetragen. Man könnte die Schleifen jedoch auch ganz anders organisieren. Es muß nicht unbedingt FOR sein...

Programmbeispiel: 10.

5. Ein kleineres Quadrat einschreiben.
Die Seitenlänge soll ein Festwert sein (z. B. 20 Pixel). Das Quadrat soll beweglich sein und an beliebiger Stelle dargestellt werden können (Basiskoordinaten).

Programmbeispiel: 11.

6. Das kleinere Quadrat über den Bildschirm bewegen.
Hierzu den Joystick 12 über den Universalapater 10b anschließen (Port A Universaladapter – Port D Starterkit). Wir verwenden den Koordinatentaster (Hat Switch). Auf die Bildränder nehmen wir keine Rücksicht.

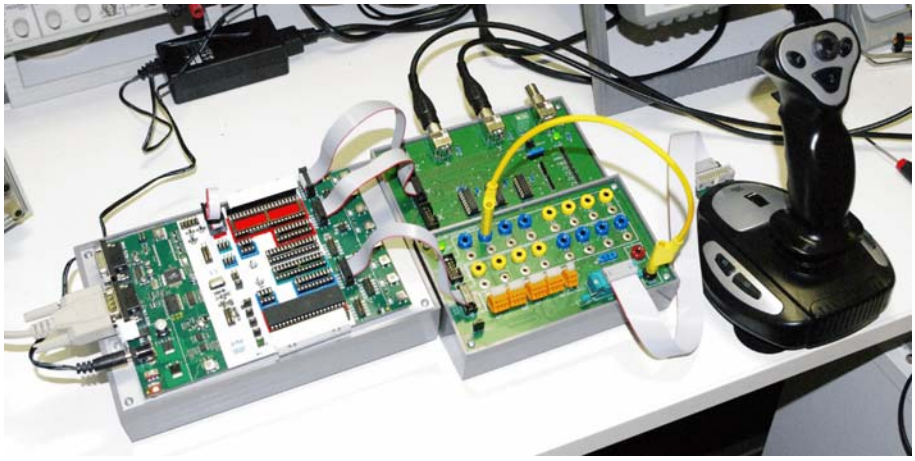


Abb. 25 Joystickanschluß.

– Port B Universaladapter –							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COM1	COM2	S4	S3	S2	Plus G	Plus Q	Plus H
Down	**Up**	**Left**	**Right**	S1 (Trigger)	Schleifer Gas	Schleifer Quer	Schleifer Höhe
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
– Port A Universaladapter –							

COM1: Potis und S5 bis S7. COM2: Hat Switch und S1 bis S4. COM2 = Port B6 (des Universalapapters) mit Masse verbinden (Laborkabel).

Programmbeispiel: 12.

Zusatzaufgabe (selbständig zu lösen)
In diesem Versuch verwenden wir nur den Port A des Universaladapters. Neben dem Koordinatentaster (Down – Up – Left – Right) ist noch der Abzugshebel (Trigger) verfügbar. Nutzen Sie dieses Bedienelement aus, um die Größe des Quadrates fortlaufend zu erhöhen. Hat es seine maximale Ausdehnung erreicht, soll es auf Größe Null zusammenfallen, dann wieder wachsen usw. (Es hört sich schwieriger an als es wirklich ist. Ausprobieren.)