1 Hello Blinking World

Student Group

First Name	Surname	Matrikel Nr.

Table of Contents

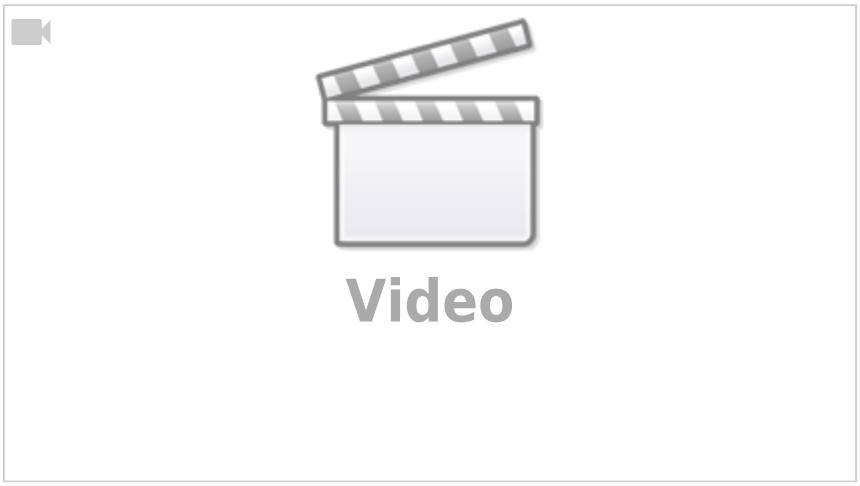
1. Hello Blinking World 2
AVR Programmierung für Dummies 2
Achtung! 2
Ziele 2
Video 2
Es werde Licht! Oder auch nicht... 2
Ziele 2
weiterführende Links 3
Video 2
Übung 3

1. Hello Blinking World

AVR Programmierung für Dummies

Video

Im Video werden im ersten Teil (bis 15min30sec) die Grundkonzepte für die AVR Programmierung erklärt.



Achtung!

Ab der 16. Minute geht es in die Assembler Programmierung. Dies ist nicht Teil des Kurses Mikroprozessortechnik.

Ziele

Nach dieser Lektion sollten Sie:

1. die verschiedenen Pins von AVR Chips kennen.
2. die verschiedenen Arten von Speichern in Chips kennen.
3. wissen was Register sind.
4. wissen für was das Spezialregister DDRx (x={A, B, C, D}) nützlich ist.

Es werde Licht! Oder auch nicht...

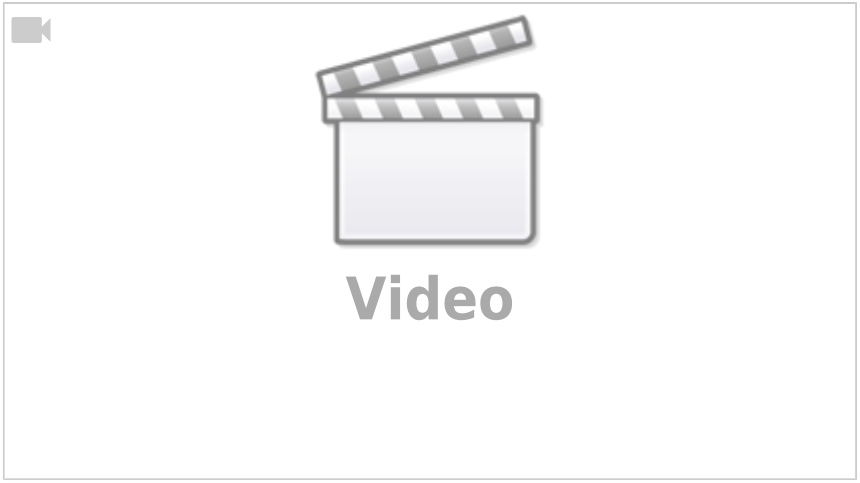
Ziele

Nach dieser Lektion sollten Sie:

1. wissen, wie man im Atmel Studio ein Projekt anlegt.
2. wissen, wie der Programmierumgebung

Video

die Taktfrequenz des Microcontrollers festgelegt wird.
3. die wichtigsten Bitmanipulationen (Bitmaske zum setzten und löschen eines einzelnen Bits, togglen) kennen und anwenden können.



weiterführende Links

- kurzes Youtube Video: ein einzelnes Bit setzen oder löschen mit bitweisen Operationen

Übung

I. Vorarbeiten

1. installieren Sie SimulIDE und Atmel Studio
2. falls es Probleme bei der Programmierung gibt: nutzen Sie die Tipps für die Fehlersuche

II. Eingabe in Atmel Studio

1. öffnen Sie Atmel Studio
2. Anlegen eines neuen Projekts
    1. `File` » New » `Project…`
    2. Wählen Sie “GCC C Executable Project”, da ein ausführbares C Code Beispiel erstellt werden soll
    3. Geben Sie bei Name `Einfuehrung_v01` und drücken Sie auf Ok
    4. Der Cursor sollte nun im Eingabefeld des Suchfenster für die Microcontroller stehen. Geben Sie dort 328 ein
    5. Wählen Sie den ATmega328 aus den möglichen Chips aus und drücken Sie auf Ok
3. Eingabe und Kompilieren des Code
    1. Ersetzen Sie den vorhandenen Code, durch den rechts stehenden Code
    2. Kompilieren Sie den Code durch `Build` » `Build solution`
    (oder dem Button “Build Solution”  oder <F7>)

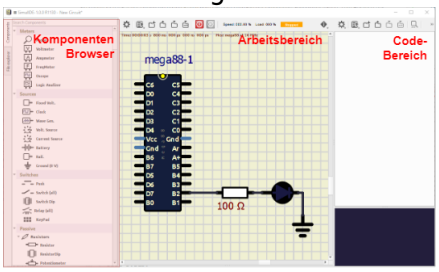
main.c

```
#define F_CPU 8000000UL

#include <avr/io.h>
#include <util/delay.h>

int main(void)
{

DDRD=0b01000000;
    while (1)
    {
        PORTD |= (1<<6);

_delay_ms(1000);
        PORTD &= ~(1<<6);

_delay_ms(1000);
    }
}
```

3. Im unteren Teil des Fensters sollte nun die Ausgabe des Kompilers sichtbar werden. Diese sollte `========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========` lauten

4. Auswählen der hex-Datei
    1. im Atmel Studio finden Sie rechts im Fenster den “Solution Explorer”
    2. gehen Sie dort im Solution Explorer zu `Solution` » `Einfuehrung_v01` » `Output Files`
    3. klicken Sie mit rechter Maustaste auf `Einfuehrung_v01.hex` und wählen Sie Pfad und Name aus

Fig. 1: einfache Diodenschaltung in SimulIDE



III. Ausführung in Simulide

1. Öffnen Sie SimulIDE (unter ...\bin\simulide.exe)
    1. links in SimulIDE sollten Sie den Komponenten Browser finden. Wählen Sie dort `Micro`»AVR»`atmega`»`atmega328`
    2. Ziehen Sie den Eintrag `atmega328` per Drag and Drop in den Arbeitsbereich (rechter, beiger Teil des Fensters)
    3. Es sollte nun ein Chip names `atmega328-1` dargestellt sein
2. Erstellen der Ausgangsschaltung
    1. Im Programm wurde im auf PortD das 6bit angesprochen. Entsprechend soll auch hier am Port D der Ausgang 6 genutzt werden. Am Chip ist dieser mit D6 gekennzeichnet
    2. Fügen Sie eine LED (im Komponenten Browser über Output LED) und ein Massepotential ein (Sources Ground)
    3. Die Komponenten können mit dem Kontextmenu (Rechtsklick) gedreht und gespiegelt werden. Außerdem ist mit der Auswahl von `Properties` im Kontextmenu die Änderung von
    4. Verbinden Sie die LED mit Masse und mit Port D6. Achten Sie auf die richtige Richtung der LED. Die Verbindungen lassen sich dadurch erstellen, dass auf ein Komponenten-Pin geklickt wird und die Linie zu einem nächsten Komponenten-Pin gezogen wird.
3. Flashen der Software
    1. Klicken Sie rechts auf den Microcontroller und wählen Sie `Load firmware`

    2. Fügen Sie hier den Pfad und Name des oben erstellten `Einfuehrung_v01.hex` ein und öffnen Sie dieses
4. Starten der Simulation
    1. klicken Sie im Menu den Power-on Button
    2. Die Simulation startet
5. Bugfixing
    1. vermutlich ist bei Ihnen zu sehen, dass die Diode nicht gleichmäßig an und aus dargestellt wird. Dies ist kein Fehler des Simulationsprogramms. Es wurde noch eine wichtige Komponente vergessen, welche immer bei der Verwendung von diskreten LEDs verwendet werden muss. Fügen Sie diese ein und Testen Sie die Schaltung nochmal

Sie sollten sich nach der Übung die ersten Kenntnisse mit dem Umgang der Umgebung angeeignet haben. Zum Festigen des der Fähigkeiten bieten sich folgende Aufgaben an:

Aufgaben

1. Welche Vorgaben für die Softwareentwicklung wurden verletzt, trotzdem das Programm lauffähig ist? (Interrupts werden erst in späteren Übungen erklärt)
2. Wie könnte ein Ampel-Licht-Abfolge oder Lauflicht aus 4 Dioden erstellt und programmiert werden? Welche Optimierungen könnten im Code vorgenommen werden? Welche Komponente in SimulIDE kann genutzt werden? Wie kann die Farbe der LEDs geändert werden?
3. Lesen Sie auf Mikrocontroller.net im Kapitel Warteschleifen die “erste Seite”, also bis:

> Abhängig von der Version der Bibliothek verhalten sich die Bibliotheksfunktionen etwas unterschiedlich.

Weiterführende Fragen und Infos

Was hat es mit "Release" und "Debug" in der Standard-Toolbar auf sich?

Mittels dieser Auswahl lassen sich verschiedene Konfigurationen des Compilers setzen. Der Compiler übersetzt den C-Code in maschinenlesbaren Code und wird bei bei Bedarf

Hinweise/Warnungen/Fehler ausgeben, diverse Optimierungen vornehmen und die ausführbaren Dateien (“Executables”) ablegen.
Über eine Konfiguration kann damit folgende Dinge geändert werden:

- Kriterien für Hinweise/Warnungen/Fehler
- Optimierungen bei der Kompilierung
- Ablageort der Dateien

Was ist DDRD, PORTD?

Die Anschlüsse (Pins) des Chips sind in 8er Gruppen sortiert, den sogenannten Ports. Für jeden Port sind jeweils drei RegisterBytes vorhanden: DDRx, PORTx und PINx. Diese Speicherstellen ermöglichen die Konfiguration des Ports. Die Bits in den Registern stehen für die einzelnen Anschlüsse: So ist zum Beispiel das 6. Bit in DDRD, PORTD und PIND für den Anschluss D6 zuständig.

Das Bit im **DDRx** (Data Direction Register) wählt die Richtung des Pins aus. Wenn dort logisch Eins geschrieben wird, wird der entsprechende Pin als Ausgangspin konfiguriert. Wenn dort logisch Null geschrieben wird, wird der entsprechende Pin als Eingangspin konfiguriert.

Das Bit im **PORTx** Register hat mehrere Eigenschaften: Wenn das gewünschte Bit in PORTx logisch eins geschrieben wird und der Pin als Ausgangspin konfiguriert ist, wird der Portpin auf high (eins) gesetzt. Wenn das gewünschte Bit in PORTx logisch Null geschrieben wird und der Pin als Ausgangspin konfiguriert ist, wird der Portpin auf Low (Null) getrieben.

Auch wenn ein Pink als Eingangspin konfiguriert wurde, hat PORTx eine Funktion. Wenn in diesem Fall das gewünschte Bit in PORTx logisch eins geschrieben wird, wird der Pull-up-Widerstand aktiviert. Ein Pull-up-Widerstand ist ein höherohmiger Widerstand (im Bereich $20 k\Omega$ ... $100 k\Omega$), der bei nicht weiter verbundenem Pin den ausgegebenen Wert auf logisch Eins zieht. Um den Pull-up-Widerstand auszuschalten, muss das gewünschte Bit in PORTx logisch Null geschrieben werden oder der Pin muss als Ausgangspin konfiguriert werden.

Das Einlesen der Signale wird in einem späteren Kapitel erklärt.

Wie findet man die Namen der Anschlüsse?

Die Namen sind im Datenblatt des verwendeten Microcontrollers zu finden. Das lässt sich in diesem Fall in einer Suchmaschine über `atmega 328 “datasheet” site:microchip.com filetype:pdf` finden, da es sich beim Datasheet um ein PDF des Herstellers Microchip handelt. Zum Lesen der Datenblätter empfiehlt sich ein Download und die Betrachtung über einen PDF-Viewer, welcher ein Inhaltsverzeichnis als Seitenleiste ermöglicht (z.B. Acrobat Reader). Ansonsten ist das Inhaltsverzeichnis häufig auch auf den hinteren Seiten des Datenblatts zu finden.

Die gesuchte Pinbelegung ist für den ATmega328P konkret auf Seite 3