Skript

Student Group

First Name	Surname	Matrikel Nr.

Table of Contents

Skript

Bildschirmlupe an!

SW2 Hello Display World - fast Counter

1. Wdh. Hello Blinking World:
    1. DDRx, PORTx, _Delay_ms() --> R steht fast immer für Register
    2. --> einmal kompilieren und in Simulide aufbauen
    3. Welche “Vorgaben für die SW-Entwicklung” wurden verletzt? --> Keine magic numbers, sondern #defines !
       siehe Weiterführende Fragen und Infos
2. Heute “Hello World” in echt! Timer + Displayausgabe
3. “Kapitel 2 Sound und Timer bitte nachträglich anschauen”
4. Frage an Studis “Wer weiß nicht was PWM ist?”

In MC Studio

1. neues Projekt `02_timer`
2. jetzt neu: mit Display!
    1. --> Bibliothek aus wiki herunterladen!
    2. Project --> Add --> existing Item (NICHT drag & drop)
    3. bei mir --> F2 Namen ändern auf `lcd_lib_de.h`
    4. Split Screen
    5. Was tun, um Lib in main einzufügen?
    6. `#include`!
        1. `#inc` + <Tab>
        2. --> Unterschied `<lib.h>` vs `"lib.h"`
3. Durchsicht der `lcd_lib_de.h`
    1. F_CPU
        1. --> CPU Frequenz, wichting für genaues Timing der delays
        2. hier `18,432 MHz` --> Minimexle Frequenz
        3. Warum `18'432'000 Hz`?
            1. ILIAS --> Elektronik Labor --> MiniMEXLE Schaltbild
            2. “Schreck!” sowiel Krams auf dem Schaltplan!
            3. Wo ist der Quarz? Quarz schwingt mechanisch im E-Feld --> Schaut im Bild aus wie ein Kondensator
    2. defines --> keine Magic numbers
    3. Funktionsprotoypen --> bitte immer am anfan angeben --> gut für eine Übersicht
4. als erstes immer Initialisierung (anlegen der Variablen, verschiedene Konfigurationen etc.)
    1. `lcd_i` + <tab>
    2. schon mal kompilieren (immer mal kompilieren zum test, ob noch alles klappt)
    3. noch nicht lauffähig, da nichts angezeigt !
5. einen String ausgeben!
    1. welche Unterfunktion wohl geeignet?
    2. Hinweis auf Inkonsistenz bei Namensgebung
    3. Eingabe `lcd_displayMessage("Hallo!", 0,0 )` --> Hinweis auf Zählanfang 0 nicht 1!
6. Flashen auf Minimexle
    1. Add Target --> STK500 --> ersten COM Port auswählen (und - falls es nicht passt - den

nächsten)
    2. Tools --> Device Programming
        1. Apply --> Device Signature sichtbar?
        2. --> Memories --> Program
7. Ausgabe von `Hallo! Zähler:`
    1. kann ä nicht schreiben , sondern schreibt µ, warum?
        1. --> Datasheet lesen!
        2. Am besten in der Schaltung den Namen suchen
        3. Googeln nach DEM16216 Datasheet --> Datenblatt etwas kurz? Blockdiagramm (immer schön Bilder in eigene Dokus machen!) --> ST7066U!
        4. Googeln nach ST7066U Datasheet
        5. Kurzes darüberscrollen über das Datasheet
        6. --> Character code Table! --> ist da ä drin? In einer schon... In der anderen is µ beim gleichen Bitmuster
        7. Also: was tun? entweder á nutzen, oder `ldc_putc(11100001);` --> was wurde vergessen? --> `%` !
    2. Vergleich in Simulide:
        1. Aufbau der Schaltung: mega88 + Hd44780 (ist kompatibel zu ST7066U)
        2. Wie verbinden? Siehe lib (wenn gut beschrieben) oder MEXLE Schaltung
        3. In lib: Port-Bits. `PIN_EN`, `PIN_RS` --> wo in Simulide?
        4. Für was steht `EN`? --> Enable. `RS` --> Register Select
        5. `PORT_DATA`: von `PORTC` nur die ersten 4 bits (0...3)
        6. in Simulide `18,432 MHz` eingeben!
        7. hex file Flashen
        8. --> animation einschalten (High/Low wird angezeigt)
        9. es wird noch nichts ausgegeben?? --> im Code schauen oder im Schaltplan!
        10. `PC0` auf `D4`, `PC1` auf `D5`, `PC2` auf `D6`, `PC3` auf `D7`
        11. jetzt klappts, aber ä an falscher Position
    3. `lcd_gotoxy` einfügen
        1. In Simulide autoload einschalten!

Jetzt: aufsteigende Zahlen ausgeben Was tun?

1. Laufvariable anlegen und nutzen: `uint8_t i=0;` und `i++` in der Schleife
2. wie gibt man Zahlen aus? `sprintf`? (kann in String einen Zahlenwert ausgeben)
3. `sprintf(output_str, "i:%03u", i);` 3 --> drei Dezimalstellen, u --> unsigned
4. `output_str` deklarieren
5. `lcd_displayMessageoutput_str, 1,0);`
6. kompilieren --> `#include <stdio.h>` vergessen
7. Simulation herunterdrehen

aktuell zählt er nicht so schnell wie die CPU kann, sondern so schnell wie er es ausgeben kann. Die CPU kann aber schneller!

1. Blick ins Datenblatt des atmega88
2. Blockbild des atmega
    1. Vergleich mit Zahnarztpraxis
    2. “Zahnarzt” macht nicht alles, sondern nur komplexere Dinge
    3. viele Helfer (Servants) die der CPU zuarbeiten
    4. PORT's unten kennen wir schon. Sind die Türsteher (doorman) für die Anschlüsse
    5. Aber auch Analog Digital Wandler, nicht flüchtiger Speicher EEPROM (non-volatile memory) und mehr

    6. Wir werden Timer und Counter nutzen
3. T/C im Inhaltsverzeichnis
    1. gibts mehrere: 8 bit TC0, 16 bit TC1 und 8 bit TC2
    2. wir nehmen 16 bit Timer/Counter und gehen zu diesem Kapitel
4. **16 bit Timer/Counter**
    1. wieder Blockbild, diesmal vom Timer / Counter
    2. wieder echt kompliziert auf den ersten Blick
5. wichtig sind immer die Register
    1. TCNTn
        1. --> timer Counter ; für was steht n? in Mathe?
        2. das ist der eigentliche aufsteigende Zählwert
    2. OCRnA --> Output Compare (“Wert zum gegen-checken”)
    3. TCCRnA --> für was steht TC? Timer Counter! Für was R? Register! --> hier neu: C für control
6. Zeilen mit `sprintf` und `lcd_displayMessage` kopieren#
7. bei Ausgabe Position ändern: `lcd_displayMessageoutput_str, 1,**7**);`
8. diesmal: `sprintf(output_str, "TC:%03u", TCNT1);` --> an zweiter Pos ausgeben. --> wichtig: 3 in 5 ändern!
    1. kompilieren und in Simulide starten
    2. TC zählt noch nicht!
    3. siehe Blockdiagramm: Control logic --> steht im folgenden in der Register Description
    4. Blick in die Tabellen, was bei Initialisierung des uC mit 0 passiert **Speziell die Register description!**
    5. bei vielen ergibt `0` normal operation
    6. aber bei CS (clock select) bedeutet `000` = keine Clock. Da kommt nix raus!
    7. also `CS10` setzen für schnellsten Zähler
    8. `TCCR1B |= 1<<CS10;`
9. jetzt zählt der Zähle echt schnell. Schneller als die Anzeige!