Hard- und Software-Engineering / Automatisierungstechnik HS1 / AU1

SS 2008

Praktikumsaufgaben

Stand: 30. 4. 08

Versuch 1

Einführung in die C-Programmierung Atmel AVR.

1. Kennenlernen des grundsätzlichen Entwicklungsganges,
2. Darstellen von Zeit mittels Software,
3. elementare Anwendungsprogrammierung.

Grundlagen:

- AVR Studio 4,
- GNU C-Compiler,
- Starterkit AVR STK 500.

Grundkonfiguration: Port C mit den LEDs verbinden. Alle Ports auf Ausgang.

Aufgabe 1: Zeitdarstellung (1). Auf Port A abwechselnd 00H und FFH ausgeben. Kontrolle: mittels Oszilloskop. Wie lange dauert die kürzeste E-A-Schleife? (Im Versuch 2 werden wir die gleiche Aufgabe mittels Assemblerprogrammierung lösen.)

Aufgabe 2: Zeitdarstellung (2). Eine Funktion MILLISEC, die n Millisekunden darstellt, wobei der Parameter n eine 16-Bit-Binärzahl ist (hierzu brauchen wir zunächst eine Schleife die eine einzige Millisekunde darstellt). Funktionskontrolle wie bei Aufgabe 1.

Aufgabe 3: Lauflicht (1). Zyklische aufeinanderfolgende Erregung der LEDs in eine Richtung.

Aufgabe 4: Lauflicht (2). Zyklische aufeinanderfolgende Erregung der LEDs abwechselnd in beiden Richtungen.

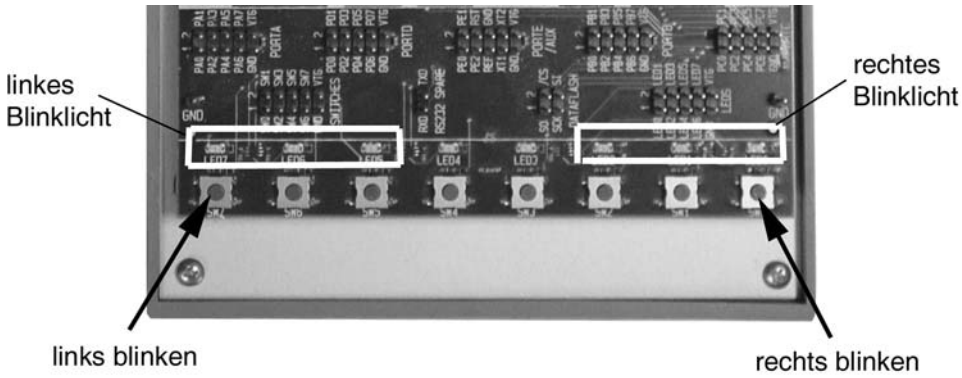
Aufgabe 5: elementare Ein- und Ausgabe. Port D mit den Tasten verbinden und auf Eingabe programmieren. Jede Betätigung einer Taste muß dazu führen, daß die entsprechende LED leuchtet.

Aufgabe 6: Kabeltester (1). Das zu prüfende Kabel wird an die Ports A und D angeschlossen. Port A auf Ausgabe (Stimulus), Port D auf Eingabe (Abfrage). Erregung auf Grundlage von Aufgabe 3. Anzeige der ankommenden Signale über Port C (LEDs).

Aufgabe 7: Kabeltester (2). Kabelanschluß wie bei Aufgabe 6. Die ankommende Belegung ist jetzt aber mit dem Stimulus zu vergleichen. Solange Übereinstimmung besteht, sind die LEDs wie bei Aufgabe 6 anzusteuern (Durchlauf). Im Fehlerfall soll die betreffende LED dauernd blinken.

Aufgabe 8: Nachblinkfunktion (vgl. beispielsweise Mercedes C-Klasse). Wird der Blinkhebel nur kurz angetippt, werden drei volle Blinkimpulse abgegeben. Bleibt der Blinkhebel länger betätigt, werden solange Blinkimpulse abgegeben, bis er wieder losgelassen wird. Wird der Blinkhebel nur kurz angetippt, aber in der Zeit, während die drei Blinkimpulse abgegeben werden, in der anderen Richtung betätigt, so wird der letzte Blinkimpuls noch zu Ende gegeben. Nach einer Impulspause wird dann in der anderen Richtung geblinkt. Wichtig: alle Blinkimpulse müssen gleich lang sein. Blinkpausen dürfen nicht kürzer sein als ein vorgegebener Mindestwert (Beispiel: Impuls und Pause je 300 ms (ausprobieren)).

Wir verwenden die Tasten und LEDs des Starterkits als Ersatz für Blinkhebel und Blinkleuchten. Anschluß LEDs: Port C, Tasten: Port D.



Versuch 2

Einführung in die Assemblerprogrammierung Atmel AVR.

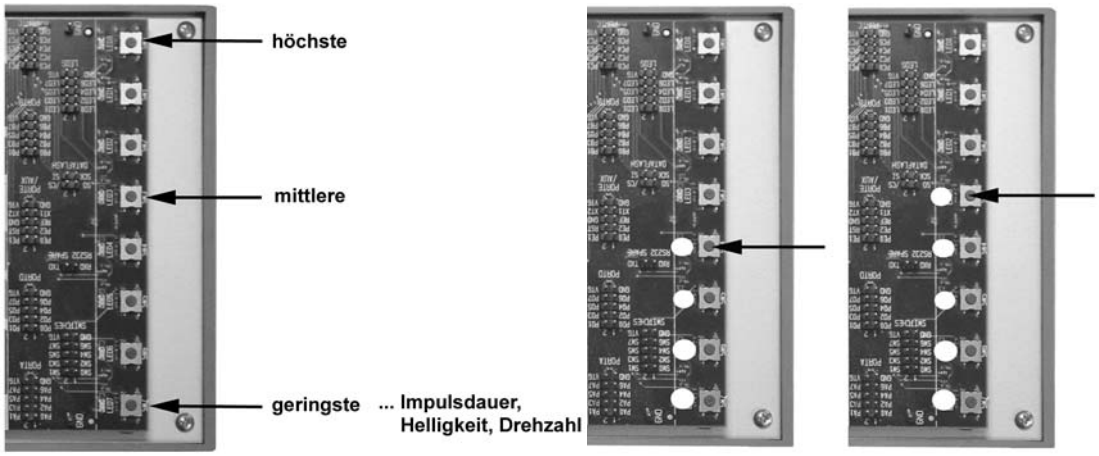
1. Kennenlernen des grundsätzlichen Entwicklungsganges,
2. Darstellen von Zeit mittels Software,
3. Arbeiten mit dem Simulator,
4. elementare Anwendungsprogrammierung.

Aufgabe 1: Zeitdarstellung (1). Auf Port A abwechslend 00H und FFH ausgeben. Kontrolle: mittels Oszilloskop. Wie lange dauert die kürzeste E-A-Schleife?

Aufgabe 2: Zeitdarstellung (2). Genaue Zeitmessung mit dem Simulator. Es sind Unterprogramme zu schreiben, um eine jeweils als Parameter zu übergebende Anzahl an Mikrosekunden und Millisekunden darzustellen.

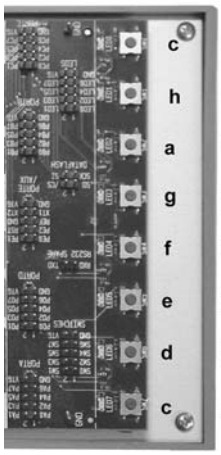
Aufgabe 3: Impulserzeugung / Pulsweitenmodulation. Mit dem Atmel-Starterkit sind variabel lange Impulse mit konstanter Periodendauer zu erzeugen (Grundlage der Pulsweitenmodulation (PWM)). Die Impulslänge soll durch die Tasten des Starterkits einstellbar sein. 1. Taste: 1/8 der Periodendauer, 2. Taste: 2/8 der Periodendauer usw. (8. Taste = volle Periodendauer = Dauersignal). Alles so einstellen (Simulator), daß die Periodendauer ungefähr 2,5 ms entspricht (400 Hz).

- Tasteneingabe über Port D.
- LED-Anzeige über Port C. Bei 1/8 Periodendauer ist nuir die erste LED an, bei 2/8 die erste und die zweite usw. (Bandskala).
- Ausgabe: über Port A. Impuls = FFH, Pause = 00H.
- Funktionsnachweis: Oszilloskop, Motoransteuerung (Experimentiertafel).



Aufgabe 4: Musikinstrument. Die Tasten werden zyklisch abgefragt. Eine betätigte Taste veranlaßt die Abgabe eines Tonsignals (Ports A und D; aktiv Low), das über Lautsprecher oder Schallgeber hörbar gemacht werden kann. Das Tonsignal ist eine symmetrische Rechteckwelle (Tastverhältnis 50:50) mit jeweils bestimmter Frequenz (gemäß chromatischer Tonleiter (s. nachstehend unter Kurzausbildung Musik; Frequenzangaben in Hz). Wenn kein Ton abgegeben wird, Ausgänge auf High (damit der angeschlossene Lautsprecher keinen Gleichstrom zieht).

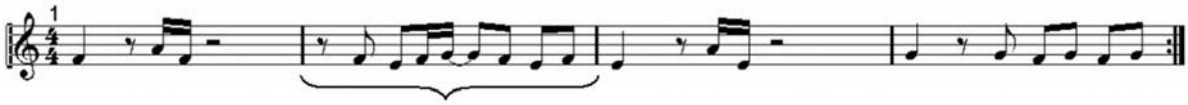
Aufgabe 5: Noten abspielen. In Fortsetzung von Aufgabe 4 (Musikinstrument) sollen Noten abgespielt werden. Die Notenfolge steht (als Zeichenkette codiert) im Programmspeicher.



	f [Hz]	$t_{p/2}$ [µs]
c	525,25	952
h	493,88	1012
a	440	1136
g	392	1276
f	349,23	1432
e	329,63	1517
d	293,67	1703
c	261,63	1911

Kurzausbildung Musik

Musik ist nicht einfach nur mit Geräusch verbunden, sondern sie reizt zum gleichnishaften Anschauen der dionysischen Allgemeinheit (F. Nietzsche); sie ist "... darin von allen anderen Künsten verschieden, daß sie nicht Abbild der Erscheinung, oder richtiger, der adäquaten Objectität des Willens, sondern unmittelbar Abbild des Willens selbst ist und als zu allem Physischen der Welt das Metaphysische, zu aller Erscheinung das Ding an sich darstellt." (A. Schopenhauer). Wir fangen allerdings erst einmal ganz klein an ...



ein Takt

Eine Taktlänge entspricht 2s unterteilt in 4 Schläge á 500ms bei einem Tempo von 120 bmp

Noten

𝅝	= ganze Note	= 4/4	=	2000 ms
𝅗𝅥	= halbe Note	= 2/4	=	1000 ms
♩	= viertel Note	= 1/4	=	500 ms
♪	= achtel Note	= 1/8	=	250 ms
𝅘𝅥𝅯	= sechzehntel Note	= 1/16	=	125 ms

Pausen — Tempo: 120 bpm

𝄻	= ganze Pause	= 4/4	=	2000 ms
𝄼	= halbe Pause	= 2/4	=	1000 ms
𝄽	= viertel Pause	= 4/4	=	500 ms
𝄾	= achtel Pause	= 4/4	=	250 ms
𝄿	= sechzehntel Pause	= 4/4	=	125 ms

andere Zeichen:

𝄇 = Wiederholungszeichen

Note	Frequenz (Hz)	Halbperiode (µs)	$^1/_{16}$ Note (125 ms) entspricht ... Perioden
c	525,25	952	66
h	493,88	1012	62
a	440	1136	55
g	392	1276	49
f	349,23	1432	44
e	329,63	1517	41
d	293,67	1703	37
c	261,63	1911	33

Codierungsbeispiel:
Jede Note wird zusammen mit der nachfolgenden Pause in 3 Bytes codiert. Eingabe als Zeichenkette (db-Direktive):

1. Notenwert (Frequenz): c, d, e, f, g, a, h, k.
2. Notendauer: 1, 2, 4, 8, 6.
3. Pausendauer: 1, 2, 4, 8, 6.

Ende der Notenkette: 0.

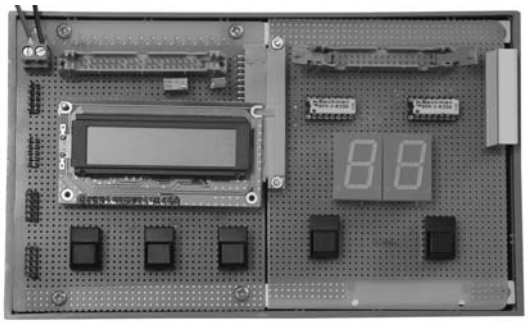
Versuch 3

Die eingebaute Peripherie (Atmel AVR) / einfache Interruptserviceroutinen.

1. LCD-Punktmatrixanzeige
2. die serielle Schnittstelle
3. der A/D-Wandler
4. Zähler und Zeitgeber

Versuchsanordnung:

- Startertkit Atmel STK 500. Ankopplung an PC über die 2. serielle Schnittstelle. Bedienung über Hyperterminal.
- Übungstafel UeSSTa 04a. Anschluß an Starterkit: Daten: Port C, Steuersignale: Port B. Einzelheiten s. Kurzbeschreibung UeSSTa 04a.
- Programmiersprache: Assembler (Atmel Developer Studio).



Aufgabe 1: Stellen Sie auf dem Dotmatrixdisplay (2 Zeilen zu 16 Zeichen) dar:

1. ein einzelnes Zeichen (Probe dafür, daß die Initialisierung geklappt hat),

2. einen einzeiligen Festtext nach eigener Wahl,

3. einen zweizeiligen Festtext anc eigener Wahl.

Bei der LCD-Ansteuerung beschränken wir uns auf reine Ausgabeabläufe, wobei die jeweilige Kommandoausführungszeit im Prozessor abgewartet wird.

Aufgabe 2: Nutzung der seriellen Schnittstelle:

1. Ausgeben von Zeichen (Festtext) auf ein Bildschirm-Terminal (Windows-Hyperterminal).

2. Schreiben Sie eine einfache Interruptserviceroutione (ISR), die das empfangene Zeichen auf den LEDs des Starterkits darstellt.

3. Erweitern Sie die ISR so, daß die empfangenen Zeichen auf der LCD-Anzeige dargestellt werden (einfachste Funktion; noch ohne Schönheiten).

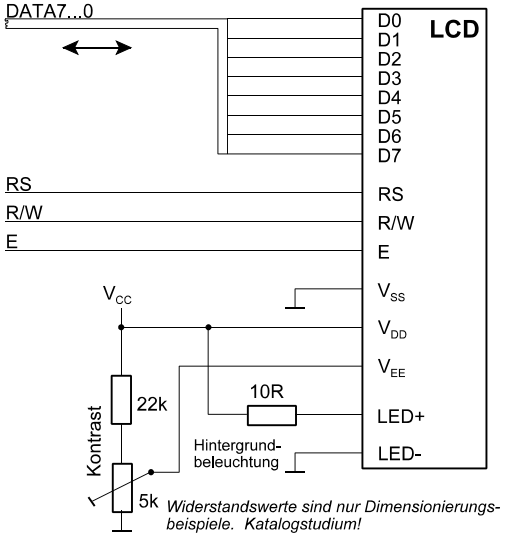
4. Erweitern Sie die ISR auf einen selbsttätigen Wrap Around (Übergang in die zweite Zeile, wenn Ende der ersten Zeile erreicht, Übergang an den Anfang, wenn Ende der 2. Zeile erreicht). Falls noch Zeit ist: die Tasten Backspace und Enter vernünftig auswerten (Rückschritt, Anfang der neuen Zeile), so daß sich die Anzeige wie ein kleines Terminal verhält.

Aufgabe 3: A/D-Wandler. Digitale Anzeige einer variablen Spannung (binär auf den LEDs des Starterkits, ggf. auch auf Bildschirm-Terminal und auf LCD).

Aufgabe 4: Pulsweitenmodulation mit Zähler/Zeitgeber. Steuerung vom Terminal aus über die serielle Schnittstelle. Z. B. Zeichen 0 = aus, + = beschleunigen, – = verringern. Nutzung als D-A-Wandler.

Kurzausbildung LCD-Display

Interface:



Datenbus: Port C, Steuersignale: Port B.

Port C:

7	6	5	4	3	2	1	0
D7	D6	D5	D4	D3	D2	D1	D0

Port B (1)
Betrifft Übungsplattform UeIDE 04 und AVR-Portadapter AVRPA 05.

7	6	5	4	3	2	1	0
ENTER	-	DOWN	ENTER	E (STB)	RW	-	RS
Eingang	x	Eingang		Ausgang		x	Ausgang

x = frei nutzbar

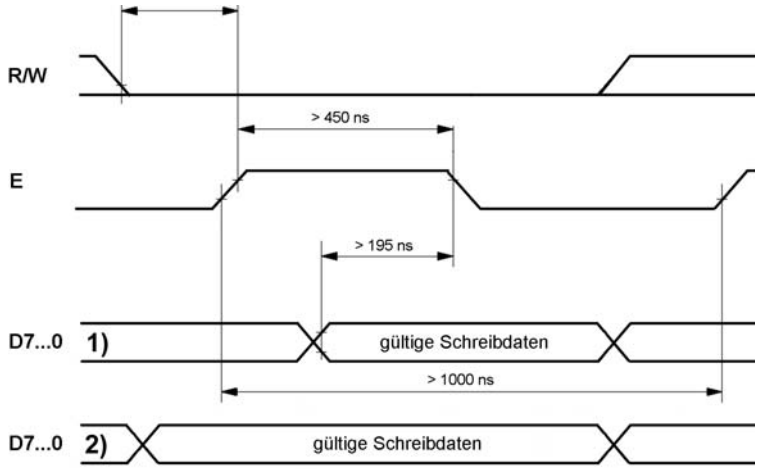
Port B (2)
Betrifft LED/LCD-Übungstafel 03a (UeLED/LCD 03a).

7	6	5	4	3	2	1	0
LED DATA	LED STROBE	LCD D/I (RS)	LCD R/W	LCD E (STB)	KEY RIGHT	KEY MIDDLE	KEY LEFT
OUT					IN		

E: LCD-Erlaubniseingang (Enable, Strobe). 0 = kein LCD-Zugriff, 1 = LCD-Zugriff.
R/W: LCD-Zugriffssteuerung. 0 = Schreiben, 1 = Lesen.
RS: LCD-Registerauswahl: 0 = LCD-Steuerregister, 1 = LCD-Datenregister.

Der typische Ablauf eines Schreibzugriffs:

1. RS je nach Zugriff einstellen. R/W auf 0.
2. Schreibdaten auf den Bus legen.
3. von Schritt 1 an müssen wenigstens 140 ns vergangen sein. E-Impuls erzeugen (mindestens 450 ns).
4. Ggf. Schreibdaten vom Bus nehmen und Bus freigeben.
5. Beginn des nächsten Zugriffs: der nächste E-Impuls darf frühestens 1 µs nach Schritt 3 ausgelöst werden.



Die Zeichendarstellung wird durch Folgen entsprechender Kommandos aufgebaut. Nach dem Senden eines Kommandos ist die Anzeige zunächst mit dessen Ausführung beschäftigt und kann kein weiteres Kommando annehmen (Besetztzustand). Wir implementieren folgenden Ablauf:

- das erste Kommando übertragen,
- lange genug warten (gemäß maximaler Ausführungszeit (s. Tabelle)),
- das zweite Kommando übertragen,
- lange genug warten usw.

Initialisierung:
Nach dem Rücksetzen Kommandos gemäß folgender Tabelle übertragen:

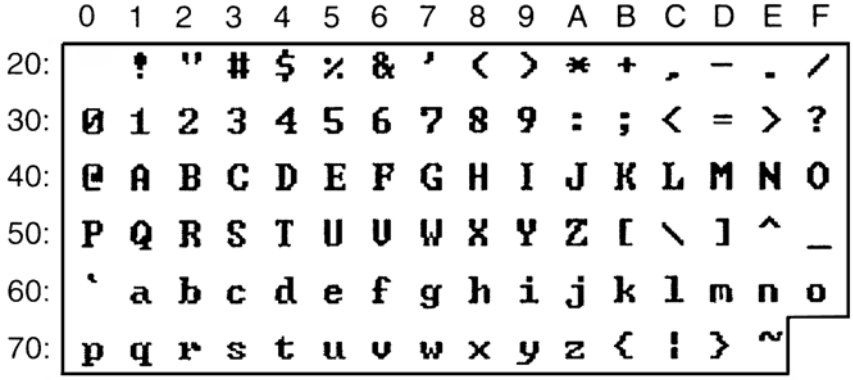
Kommando	Steuerl.		Datenbyte								Anmerkungen
	RS	R/W	7	6	5	4	3	2	1	0	
Function Set	0	0	0	0	1	1	1	0	0	0	8-Bit-Betrieb. 2 Zeilen, Zeichenraster 5 • 8
Clear Display	0	0	0	0	0	0	0	0	0	1	Datenspeicher löschen. Cursor auf erste Zeichenposition (links (oben))
Display On/Off Control	0	0	0	0	0	0	1	1	1	1	Anzeige ein, Cursordarstellung ein, Cursor blinken
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	Cursoradresse zählt aufwärts (Autoincrement); kein Schieben

Kommandoübersicht:

Kommando	Steuerl.		Datenbyte								Beschreibung	max. Ausführungszeit[2)]
	RS	R/W	7	6	5	4	3	2	1	0		
Clear Display	0	0	0	0	0	0	0	0	0	1	gesamte Anzeige löschen. Datenspeicheradresse auf Null	1,52 / 1,64 ms
Return Home	0	0	0	0	0	0	0	0	1	*[1)]	Datenspeicheradresse auf Null. Anzeige an Originalposition (keine Verschiebung). Datenspeicherinhalt bleibt erhalten	1,52 / 1,64 ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Einstellung der Bewegungsrichtung des Cursors (I/D). Wahl zwischen Cursorbewegung und Verschieben der Anzeige (S)	37 / 40 µs
Display On/Off Control	0	0	0	0	0	0	1	D	C	B	Anzeige ein/aus (D), Cursor ein/aus (C), Blinken an Cursorposition ein/aus (B)	37 / 40 µs
Cursor/Display Shift	0	0	0	0	0	1	S/C	R/L	*[1)]	*[1)]	Cursorbewegung und Verschieben der Anzeige	37 / 40 µs
Function Set	0	0	0	0	1	DL	N	F	*[1)]	*[1)]	Einstellung der Zugriffsbreite (DL), der Zeilenzahl (N) und des Zeichensatzes (F)	37 / 40 µs
CG RAM Adrs Set	0	0	0	1	Zeichengeneratoradresse						Adresse des ladbaren Zeichengenerators einstellen	37 / 40 µs
DD RAM Adrs Set	0	0	1	Datenspeicheradresse							Datenspeicheradresse einstellen	37 / 40 µs
Busy Flag / Adrs Read	0	1	BF	Adreßzähler							Busy-Bit (BF) und aktuelle Adreßzählerbelegung lesen	-
CG RAM / DD RAM Data Write	1	0	zu schreibendes Byte								Schreiben in ausgewählten Speicher	37 / 40 µs
CG RAM / DD RAM Data Write	1	1	gelesenes Byte								Lesen des ausgewählten Speichers	37 / 40 µs

1): bedeutungslos (don‘t care); 2): 44780U / herkömmliche Ausführungen

Zeichentabelle (ASCII):



Datenspeicheradressierung:

Anzeige	1. Zeile	2. Zeile	3. Zeile	4. Zeile
1 • 8	00H...07H			
1 • 16	00H...0FH			
1 • 16 (8 + 8)	00H...07H, 40H...47H			
1 • 20	00H...13H			
1 • 40	00H...27H			
2 • 8	00H...07H	40H...47H		
2 • 12	00H...0BH	40H...4BH		
2 • 16	00H...0FH	40H...4FH		
2 • 20	00H...13H	40H...53H		
2 • 24	00H...17H	40H...57H		
2 • 40	00H...27H	40H...67H		
4 • 16	00H...0FH	40H...4FH	10H...1FH	50H...5FH
4 • 20	00H...13H	40H...53H	14H...27H	54H...67H

Es gibt keinen automatischen Übergang von Zeile zu Zeile. Vor dem Eintragen in eine bestimmte Zeile jeweils Datenspeicheradresse setzen (Kommando *DD RAM Adrs Set*).