Praktikum Mikrocontrollertechnik SS 2014
Versuch 3

Stand: 28. 5. 2014

Elementare Anwendungs- umd Schnittstellenprogrammierung in Assembler unter Einschluß von Interruptserviceroutinen (ISRs).

1. Inbetriebnahme der seriellen Schnittstelle mittels Terminalprogramm.
2. Inbetriebnahme einer LCD-Punktmatrixanzeige.
3. Inbetriebnahme einer Zähler-Zeitgeber-Einheit zwecks Pulsweitenmodulation (PWM).

Versuchsanordnung:
Startertkit Atmel STK 500 mit Atmel ATMega 16. Peripherie: PC mit Terminalprogramm, LCD-Anzeige 09 mit Punktmatrix-LCD 2 • 16, Lüftertrainer 14a.

- Auf dem Starterkit Port C mit den LEDs verbinden.
- Das Projekt des zweiten Versuchs aufrufen oder eine neues Assemblerprojekt einrichten.
- Das Programm **V3_14_template.asm** in das Editorfenster des Projekts kopieren.
- Die Datei **V3_14_examples.asm** im Editor öffnen, um ggf. Programmbeispiele übernehmen zu können.
- Das Projekt implementieren (Build + Programmieren), um den Programmer im AVR Studio zu veranlassen, seine serielle Schnittstelle zu reservieren.
- Funktionskontrolle: das Musterprogramm enthält eine Lauflichtfunktion. Es muß ein zyklisch hin- und herlaufendem Leuchtpunkt zu sehen sein.

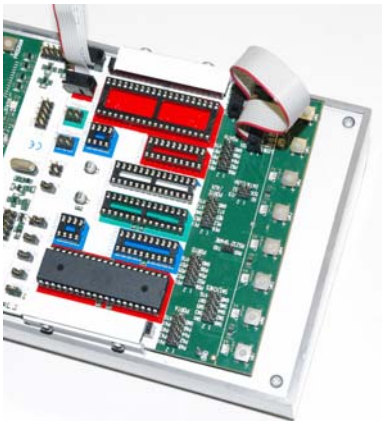


Abb. 1 Der erste Versuchsaufbau. Die LEDs sind an Port C angeschlossen.

Aufgabe 1: Serielle Schnittstelle.

Kurzausbildung serielle Schnittstelle
Die serielle Schnittstelle ist ein bitserielles asynchrones Interface. Je Richtung ist eine Datenleitung vorgesehen. Es werden einzelne Zeichen übertragen (Start-Stop-Verfahren). Der Übertragung liegt ein festes Zeitraster (Übertragungsrate) zugrunde. Die Zeichen werden mit zusätzlichen Start- und Stopbits voneinander abgegrenzt. Erkennt der Empfänger ein Startbit, so beginnt er, den ankommenden Datenstrom mit seinem Takt abzutasten. Das Abtasten endet mit dem Empfang des (bzw. der) Stopbits. Danach wartet der Empfänger auf das nächste Startbit. Codierung der Bitfolgen: NRZ. Übertragungsreihenfolge: von der niedrigstwertigen zur höchstwertigen Bitposition (LSB => MSB).

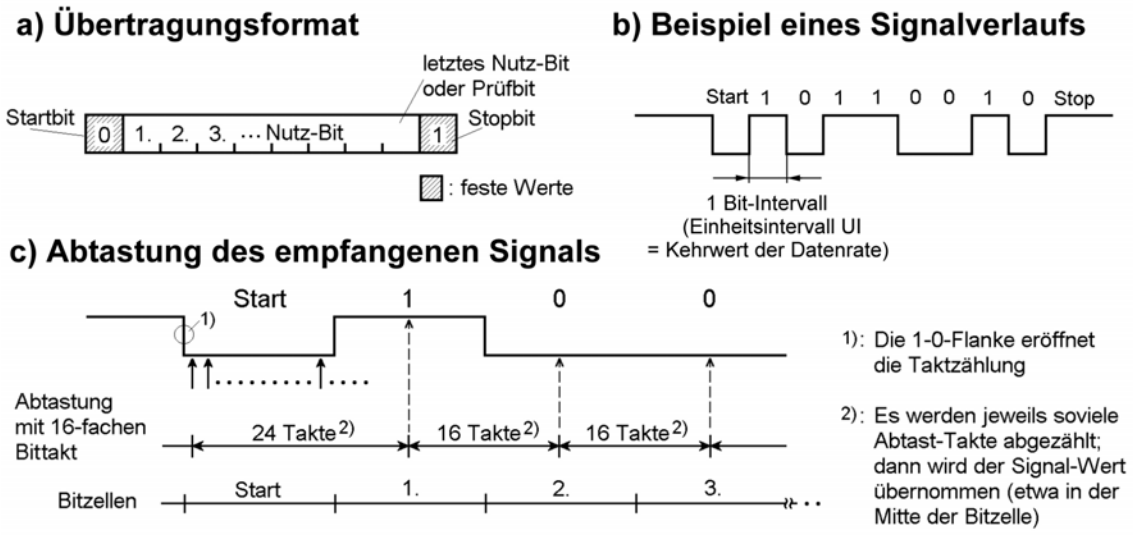


Abb. 2 Das Übertragungsprinzip.

Der Ruhezustand wird durch einen Eins-Pegel signalisiert. Die Übertragung eines Zeichens beginnt mit einem Nullbit (Startbit). Der erste Eins-Null-Übergang – aus dem Ruhezustand heraus – veranlaßt den Empfänger, mit dem Abtasten des ankommenden Signals zu beginnen. Jede Bitzelle wird mehrmals abgetastet, beispielsweise mit einem Takt, der die 16fache Frequenz des Bittaktes hat. Trifft der erste Abtastimpuls auf den Eins-Null-Übergang, so hat man nach weiteren 24 solchen Impulsen ziemlich sicher die Mitte der nachfolgenden Bitzelle getroffen. Diese enthält das erste Nutz-Bit des übertragenen Zeichens. Mit jeweils 16 Abtastimpulsen Abstand werden dann die weiteren Bitzellen näherungsweise in der Mitte abgetastet. Sind alle Zeichenbits (und ggf. ein zusätzliches Paritätsbit) übertragen worden, wird ein Einsbit als Endekennung (Stopbit) erwartet. Kommt keine 1, liegt ein Fehler vor (Fachbegriff: Framing Error). Daraufhin gelangt der Empfänger in den Ruhezustand und erwartet das nächste Startbit. Es gibt auch Übertragungsformate mit 2 oder mit 1½ Stopbits ("1½ Bits" bedeutet, daß der High-Pegel wenigstens 1½ Bitzellen lang anliegen muß).

Gängige Übertragungsraten (in Bits/s):

50	75	110	134,5	150	200	300
600	1200	1800	2000	**2400**	3600	4800
7200	9600	14400	19200	38400	57600	115200

Die serielle Schnittstelle im ATMega16

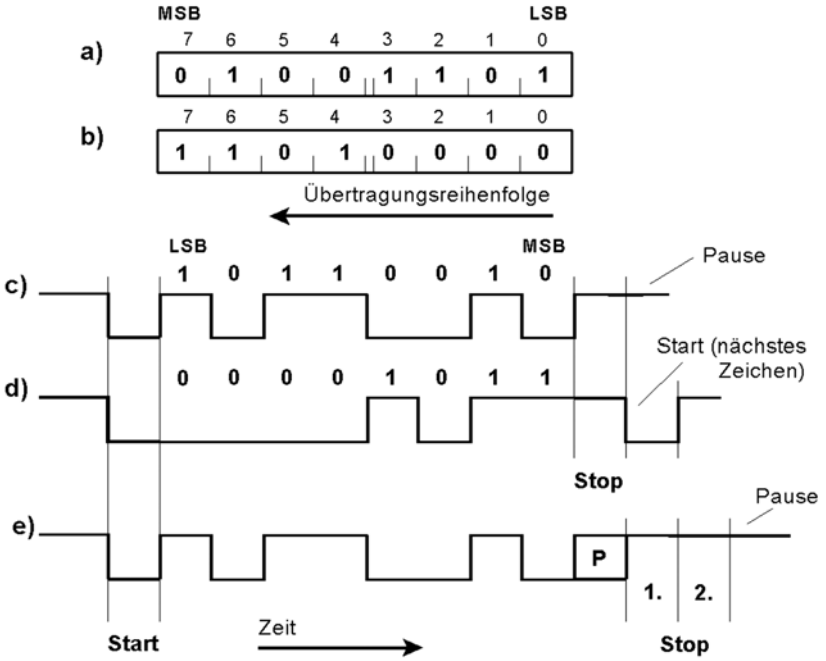
Wir beschränken uns auf die einfachsten Betriebsarten. Manche Einstellungen ergeben sich bereits beim Rücksetzen.

Programmseitige Steuerung:

a) Einstellung der Baudrate: UART Baud Rate Register UBRRH und UBRRL.
b) Betriebsartensteuerung und Zustandsabfrage: UART Control and Status Register A, B, C (UCSRA, UCSRB, UCSRC).
c) Datenbytes eintragen oder abholen: UART I/O Data Register UDR.

Wir arbeiten mit folgenden Einstellungen:

- Baudrate 2400,
- 8 Datenbits,
- kein Paritätsbit,
- 1 Stopbit,
- Senden über Abfrage,
- Empfangen über Unterbrechung.



a), b) Zwei zu übertragende Bytes (Beispiele). Die Übertragung beginnt stets mit dem niedrigstwertigen Bit (LSB).

c) Übertragung von Byte a). 10-Bit-Format. Keine Parität, 1 Stopbit. Pause nach Übertragung des Zeichens.

d) Übertragung von Byte b). 10-Bit-Format. Keine Parität, 1 Stopbit. Nach dem Stopbit folgt sofort das Startbit des nächsten Zeichens (schnellste Übertragungsfolge).

e) Übertragung von Byte a). 12-Bit-Format. Paritätsbit (P), 2 Stopbits. Pause nach Übertragung des Zeichens. Wert des Paritätsbits P hängt von programmseitiger Einstellung im UART ab. Im Beispiel werden 4 Einsen übertragen. Deshalb ist P bei gerader Parität = 0, bei ungerader = 1.

Abb. 3 Übertragungsbeispiele.

Baudrateneinstellung:
Die Baudrate ergibt sich gemäß Formel:.

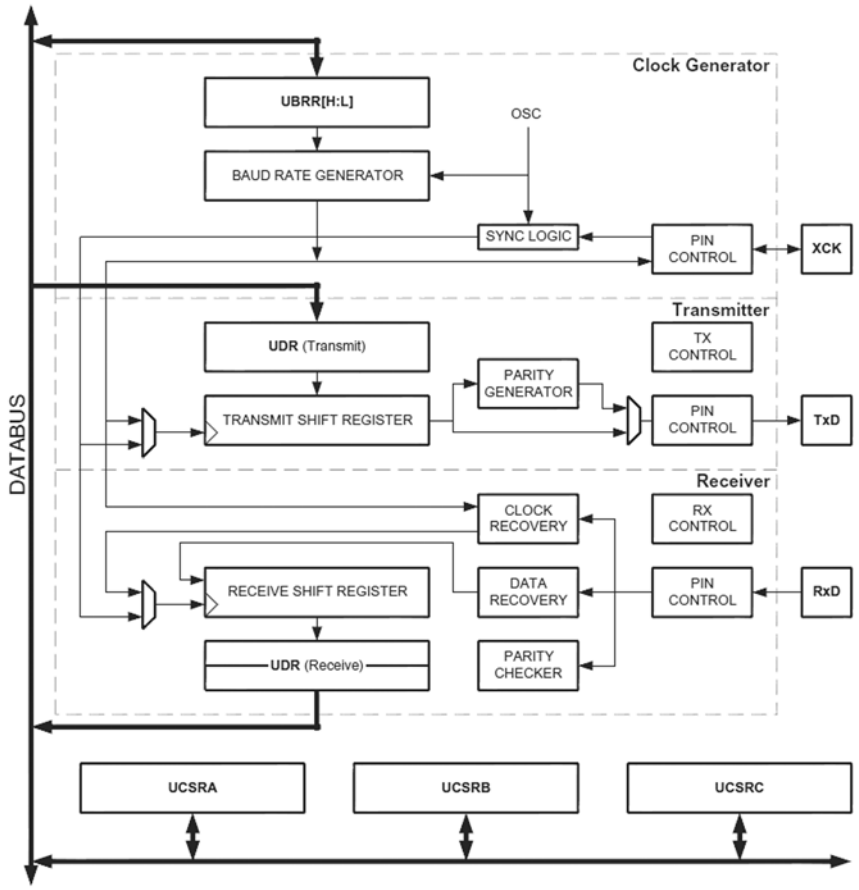
$$\mathrm{BAUD} = \frac{f_C}{16 \cdot (\mathrm{UBRR} + 1)}$$

$$\mathrm{UBRR} = \frac{f_C}{16 \cdot \mathrm{BAUD}} - 1$$

f_C = Taktfrequenz.

Typische UBRR-Werte können aus Tabellen im Datenbuch entnommen werden. UBRR ist als Binärzahl von 12 Bits Länge einzustellen; 4 Bits in UBRRH, 8 Bits in UBRRL.

Für 2400 Baud (= Bits/s) brauchen wir bei f_C = 4 MHz nur das niederwertige Byte. Wert = 103.



(Bildquelle: Atmel.)

Abb. 4 Blockschaltbild der Schniutstellenhardware. UART = Universal Asynchronous Receiver/Transmitter.

UART Baud Rate Register Low (UBRRL)

7	6	5	4	3	2	1	0
UBRRL7							UBRRL0

UART Status and Control Register A (USRA)

7	6	5	4	3	2	1	0
RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM

- RXC: Zeichen empfangen.
- TXC: Zeichen gesendet.
- UDRE: Datenregister leer.
- FE: Stopbit des empfangenen Zeichens = 0 (Framing Error).
- DOR: Überlauf. Ein empfangenes Byte ist noch anhängig (nicht abgeholt worden), und es kommt schon eine neues (Data Overrun).
- PE: Paritätsfehler.
- U2X: Doppelte Übertragungsgeschwindigkeit.
- MPCM: Multiprozessor-Kommunikationsmodus. Auswertung des 9. Datenbits als Adresse.

Wir fragen UDRE ab, um zu erfahren, ob das Sendedatenregister frei ist oder nicht. Wurde UDRE als gesetzt vorgefunden, müssen wir es wieder löschen. Hierzu ist eine Eins in diese Bitposition zu schreiben.

UART Control and Status Register B (UCSRB)

7	6	5	4	3	2	1	0
RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8

- RXCIE: Interruptauslösung nach Empfang eines Zeichens.
- TXCIE: Interruptauslösung nach Senden eines Zeichens.
- UDRIE: Interruptauslösung, wenn Datenregister leer (beim Senden).
- RXEN: Empfang ein- und ausschalten. 0 = ausgeschaltet, 1 = eingeschaltet.
- TXEN: Sendebetrieb ein- und ausschalten. 0 = ausgeschaltet, 1 = eingeschaltet.
- UCSZ2: eine Bitstelle der Zeichenlängeneinstellung.
- RXB8: das 9. Bit des empfangenen Zeichens.
- TXB8: das 9. Bit des zu sendenden Zeichens.

Wir setzen: RXCIE, RXEN und TXEN, also UCSRB = 98H.

Das UART Control and Status Register C (UCSRC) ist schon vom Rücksetzen her richtig eingestellt.

UART I/O Data Register (UDR)

7	6	5	4	3	2	1	0
D7							D0

Es sind zwei Register unter einer Adresse. Das Schreiben (Ausgabe) betrifft das Sendedatenregister, das Lesen (Eingabe) das Empfangsdatenregister.

Versuchsaufbau:

1. Die serielle Erprobungsschnittstelle des Starterkits mit Port D, Bits 0 und 1, verbinden (Brückenkabel).
2. Das zweite Schnittstellenkabel des PCs ans Starterkit anschließen.
3. Terminalprogramm aufrufen. Es ein ganz einfaches Programm, an dem nichts einzustellen ist.
4. Den COM-Port auswählen. Es ist entweder COM-Port 1 oder COM-Port 2. Wenn es nicht klappt, mit dem jeweils anderen COM-Port probieren oder das andere Schnittstellenkabel anschließen.
5. Programm eingeben und zum Laufen bringen.

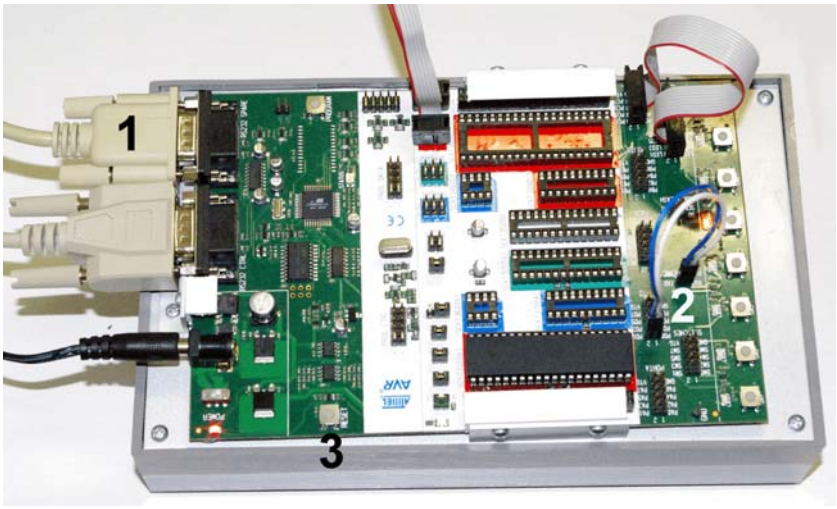


Abb. 5 So wird die serielle Schnittstelle angeschlossen. 1 - Kabel zum PC; 2 - Verbindung des Treiberschaltkreises mit Port D (Signale PD0 und PD1); 3 - RESET-Taste.

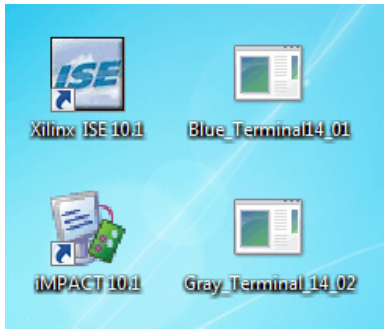


Abb. 6 Ein Terminalprogramm auswählen.

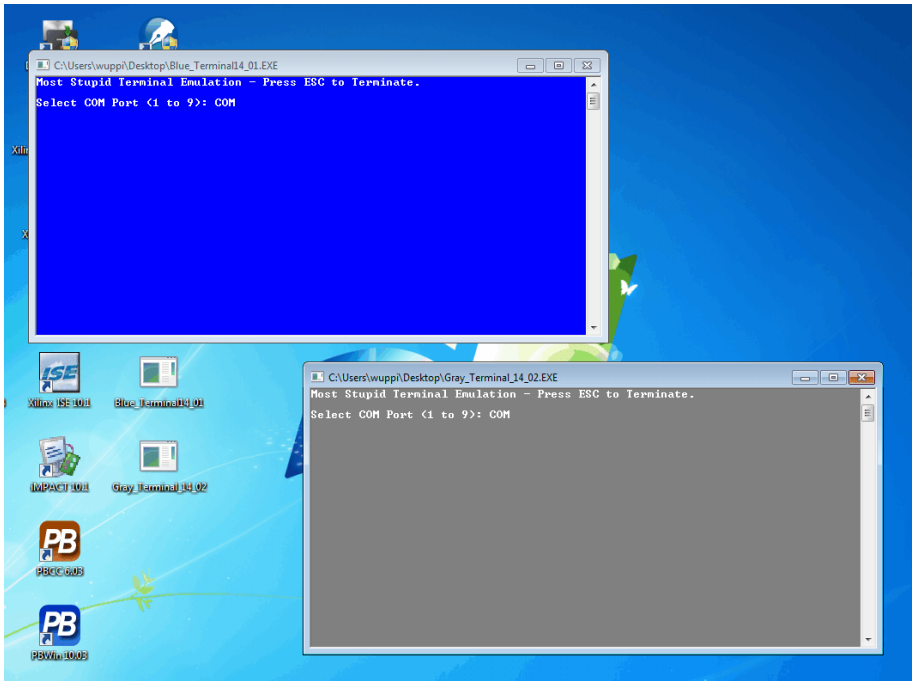


Abb. 7 Die Wahl ist reine Geschmackssache. Die Nutzung erklärt sich von selbst.

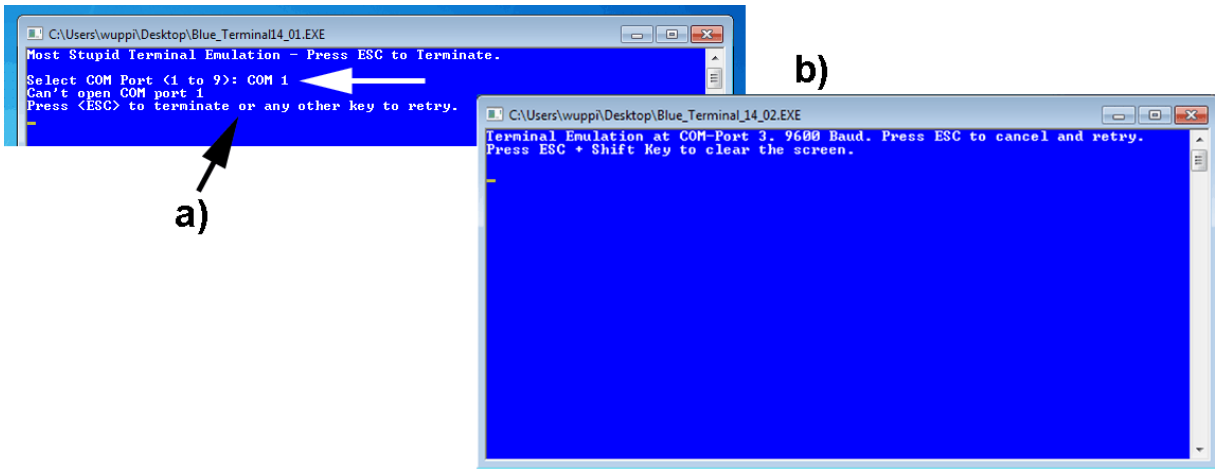


Abb. 8 Zur Auswahl des COM-Ports die Nummer (1 oder 2) eingeben (Pfeil). a) erscheint, wenn es nicht klappt, b) wenn alles in Ordnung ist.

Eine kleine Fehlersuchhilfe
Wenn im Terminalfenster nichts zu sehen ist:

- Richtiges Schnittstellenkabel angesteckt?
- Schnittstelle auf dem Starterkit richtig angeschlossen?
- Ggf. nachmessen (Oszilloskop).

Abb. 9 Ein Praxistip zwischendurch: Welches Kabel gehört zu welchem COM-Port? Das kann man mit einem Kurzschlußstecker experimentell herausfinden. Die Signale TX und RX (Pins 2und 3) sind hier miteinander verbunden (echte Profis schaffen es auch mit einem Schraubenzieher oder einer aufgebogenen Büroklammer...). Gehört das Kabel zum ausgewählten COM-Port, werden Tastatureingaben auf dem Terminalfenster angezeigt (Echofunktion). Kurzschlußstecker kann bei Bedarf ausgeliehen werden.

Bedienhinweise:
- Den Bildschirm löschen: Die Umschalttaste (SHIFT) niederhalten und ESC betätigen.
- Einen COM-Port zuweisen: ESC betätigen.
- Das laufende Programm beenden: ESC zweimal betätigen.

1. Die Schnittstelle im AVR initialisieren.
Baudrateneinstellung = 103. Ein einzelnes Zeichen ausgeben (zu Probe, ob die Initialisierung geklappt hat). Ggf. die RESET-Taste am Starterkit betätigen, um das Senden erneut auszulösen.

Programmbeispiel: 1.

2. Einen Text (im Flash gespeicherte Zeichenkette) ausgeben.
Die Zeichenkette wird mit 00H abgeschlossen (vgl. Programmiersprache C). Welche Zeichen können angezeigt werden? – Terminalcodes sind eine Wissenschaft für sich (s. die einschlägigen Handbücher und Hilfedateien). Ohne weiteres verfügbar sind nur das lateinische Alphabet ohne Umlaute und Eszett, die die Ziffern und die übliche Satzzeichen – mit anderen Worten, der Zeichensatz des sog. ASCII-Codes. Codetabelle am Ende dieser Versuchsanleitung.

Programmbeispiel: 2.

3. Interruptserviceroutine zum Empfangen von Zeichen.
Das jeweils empfangene Zeichen über Port C auf die LEDs des Starterkits ausgeben.

Programmbeispiel: 3.

4. Echofunktion
Die empfangenen Zeichen zum PC zurücksenden, so daß sie auf dem Bildschirm so erscheinen, als wären sie an Ort und Stelle eingetippt worden.

Programmbeispiel: 4.

5. Die Echofunktion auf die Unterstützung der Enter-und der Backspace-Taste (BS = 08) erweitern.
Tastencodes: Enter-Taste (CR) = 13; Backspace-Taste (BS) = 08. Die Zeichenausgabe zum Port C entfernen.

- Echo bei Enter: Rücklauf und neue Zeile (CR LF = 13 10).
- Echo bei Backspace: Löschen des letzten Zeichens und Zurücksetzen des Cursors (Folge BS – SP – BS = 08 32 08).

Programmbeispiel: 5.

Aufgabe 2: LCD-Punktmatrixanzeige.
Die LCD-Anzeige 09 anschließen. Daten an Port C, Steuersignale an Port B.

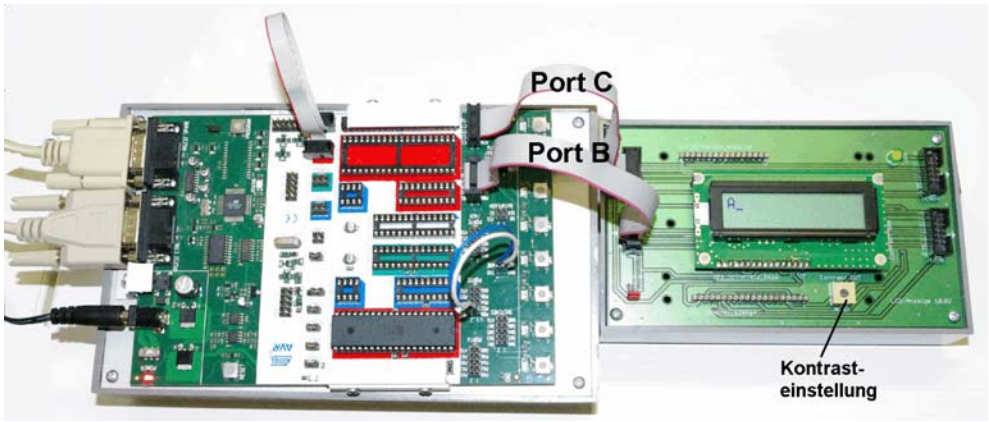


Abb. 10 Versuchsaufbau mit LCD-Anzeige. Die serielle Schnittstelle bleibt angeschlossen.

Kurzausbildung LCD-Punktmatrixanzeige (Dotmatrix Display)
Punktmatrixanzeigen dienen zum Darstellen von Zeichen. Eine Anzeige des Typs X • Y kann X Zeilen zu jeweils Y Zeichen darstellen. In unserem Versuchaufbau ist die Anzeigeeinheit mit einem Anzeigemodul 2 • 16 bestückt.

Datenbus: Port C

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

Steuersignale: Port B

7	6	5	4	3	2	1	0
–	–	–	–	–	E1	R/W	RS

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.

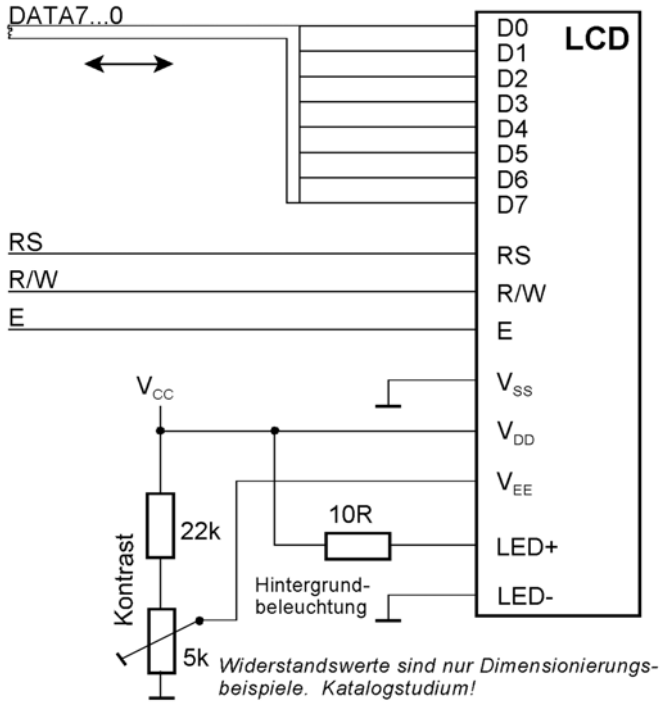


Abb. 11 Die Schnittstelle einer typischen Punktmatrixanzeige. 8-Bit-Datenbus, drei Steuersignale.

Es ist nicht zwingend erforderlich, Lesezugriffe zu unterstützen. Schreibzugriffe genügen vollauf. Wir beschränken uns deshalb auf diese Betriebsweise.

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. Er muß mindestens 450 ns lang sein.
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.

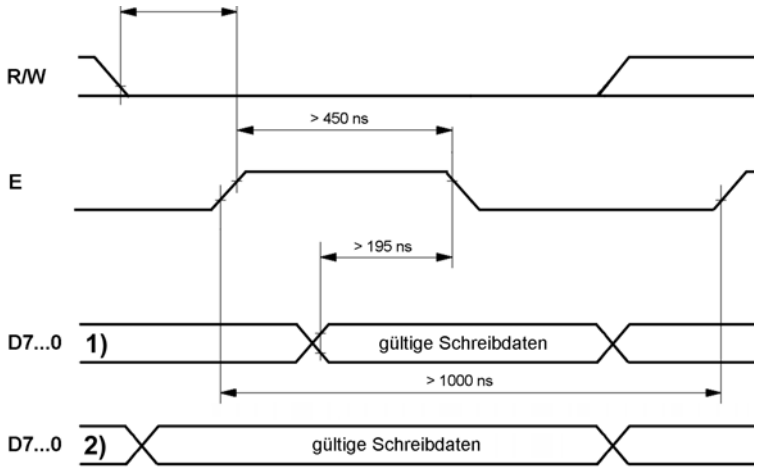


Abb. 12 Ein Schreibzyklus.

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 laut Kommandoübersicht).
- Das zweite Kommando übertragen.
- Lange genug warten usw.

Darstellbare Zeichen:
Es gibt verschiedene Zeichensätze. Zu den Einzelheiten siehe das Datenmaterial der Anzeigemodule. Grundsätzlich verfügbar sind – wie beim Terminal – nur die Zeichen des ASCII-Codes. Weitere Darstellversuche (Umlaute, Eszett, griechisch, kyrillisch usw.) sind ohne Datenblattstudium zwecklos.

Initialisierung:
Datenbits und Steuersignale auf Ausgabe. 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

Datenspeicheradressierung:

Anzeige	**1. Zeile**	**2. Zeile**	**3. Zeile**	**4. Zeile**
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*).

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.

1. LCD initialisieren.

Wir arbeiten mit Zeitsteuerung (Schreiben und Ausführungszeit abwarten; kein Zurücklesen). Ein einzelnes Zeichen darstellen (zur Probe, ob die Initialisierung geklappt hat). Zeichen muß links oben erscheinen, rechts daneben muß der Cursor blinken. Ggf. Kontrast nachstellen (Trimmpotentiometer).

Programmbeispiel: 6.

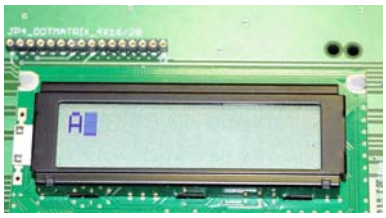


Abb. 13 Die erste Zeichenanzeige.

2. Einen einzeiligen Festtext darstellen.
Wir nehmen die Zeichenkette, die wir auch zum Terminal übertragen haben. Der Ablauf ist ähnlich.

Programmbeispiel: 7.

3. Einen zweizeiligen Festtext darstellen.

Programmbeispiel: 8.

4. Die Anzeige zyklisch umlaufen lassen (Laufschrift).
Hierzu nutzen wir das Verschiebekommando der LCD-Anzeige. Kommandocode = 18H.

Programmbeispiel: 9.

So besonders sieht es nicht aus. Das liegt daran, daß der gesamte Anzeigepufferinhalt verschoben wird. Es sind immer 80 Zeichen, die umlaufen, bei einer zweizeiligen Anzeige also 2 • 40. Sie können versuchen, eigene Texte[1] zu gestalten, die unter dieser Bedingung einigermaßen ansehnlich sind.

5. Die Interruptserviceroutine (ISR) der seriellen Schnittstelle erweitern
Und zwar so, daß die empfangenen Zeichen auf der LCD-Anzeige dargestellt werden. Zunächst probieren wir es einfach und schmucklos.

Programmbeispiel: 10.

6. Die ISR auf selbsttätigen Wrap Around erweitern.
Das heißt: Übergang in die zweite Zeile, wenn Ende der ersten Zeile erreicht, Übergang an den Anfang, wenn Ende der 2. Zeile erreicht.

Programmbeispiel: 11.

7. Die Tasten ENTER und BACKSPACE vernünftig unterstützen.
Die Anzeige soll sich wie ein kleines Terminal verhalten. ENTER führt an den Anfang der nächsten Zeile. BACKSPACE bewegt den Cursor um eine Position nach links (Rückschritt). Sowohl die bisherige als auch die neue Zeichenposition werden gelöscht. Löschen heißt Eintragen eines Leerzeichens (SPACE = 20H). Praxistip: Erst einmal die ENTER-Funktion allein (einfacher) abprogrammieren, dann BACKSPACE hinzunehmen.

Programmbeispiele: 12 und 13.

Aufgabe 3: Pulsweitenmodulation (PWM).

Kurzausbildung Pulsweitenmodulation
Die Pulsweitenmodulation dient dazu, kontinuierliche Größen, wie die Helligkeit einer Lichtquelle, die Drehzahl eines Motors oder die Amplitude einer Spannung, auf rein digitale Weise einzustellen, also durch einfaches Ein- und Ausschalten. Hierzu werden Impulsfolgen mit fester Periodendauer, aber veränderlicher Breite abgegeben. Je breiter der Impuls, desto höher die Helligkeit, Drehzahl usw. Gar kein Impuls ergibt den Aus-Zustand. Eine ständige Erregung (Dauersignal) ergibt den jeweiligen Maximalwert der Helligkeit, Drehzahl usw. ("Vollgas"). Die kontinuierliche Wirkung bei impulsweiser Erregung ergibt sich infolge von Integrationswirkungen. Bei Lichtquellen integriert das Auge des Betrachters (eine entsprechend angesteuerte LED flimmert im Takt der Erregung, wir können das aber nicht wahrnehmen), beim Motor ist es die mechanische Trägheit.

1: Aber bitte nach Möglichkeit nicht allzu primitive, ordinäre usw.

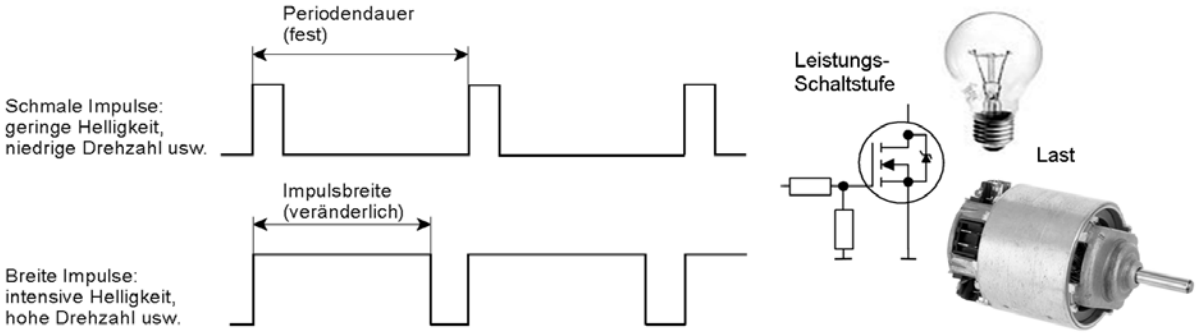


Abb. 14 Zum Prinzip der Pulsweitenmodulation (PWM).

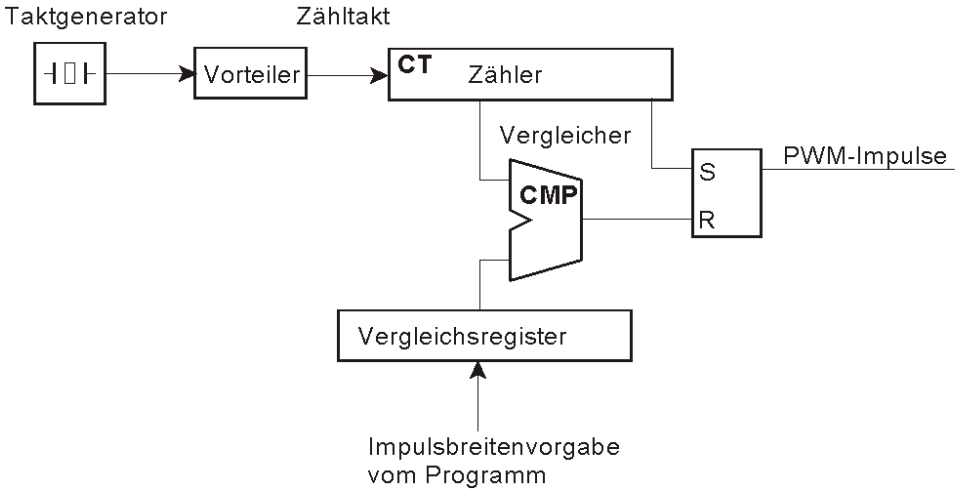


Abb. 15 Das Prinzip der Impulserzeugung. Der Periodenzähler läuft ständig um. Läuft er über das Zählende hinaus wieder in die erste Stellung ein, setzt er das Flipflop. Der PWM-Impuls wird aktiv. Entspricht der Zählerstand dem Wert der vorgegebenen Impulsbreite, so setzt der Vergleicher das Flipflop wieder zurück. Der Impuls ist damit zu Ende.

Unser Mikrocontroller soll solche Impulse abgeben. Wir verwenden die Zähler-Zeitgeber-Einheit (Counter/Timer) 0. Deren Ausgang: OC0 = Port B Bit 3.

Die Wirkungsweise der Schaltung veranschaulichen wir uns mittels Oszilloskop, die anwendungspraktische Wirkung der Pulsweitenmodulation durch Anschließen des Lüftertrainers 14a. Die LCD-Anzeige abbauen. Die serielle Schnittstelle bleibt. Zunächst nur das Oszilloskop anschließen und die Schaltung ausprobieren.

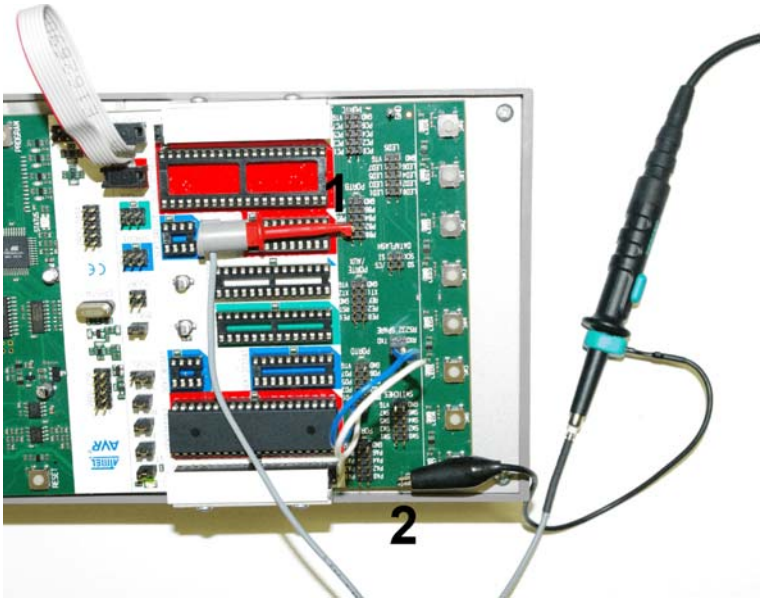


Abb. 16 So wird das Oszilloskop angeschlossen: 1 - der Haken des Tastkopfes an PD3 (Port B). 2 - die Masseklemme an einen Meßpunkt GND.

Die aktuelle Zählweite wird in einem Universalregister gehalten. Wir beginnen mit einem mittleren Wert (128). Um die Zählweite zu verändern, nutzen wir das Terminal, die serielle Schnittstelle und die Interruptserviceroutine. Z. B. Zeichen 0 = aus (Wert 0) , + = beschleunigen, – = verringern.

Programmseitige Steuerung:

a) Einstellen der Betriebsart: Timer/Counter Control Register TCCR0.
b) Laden des Vergleichswertes: Output Compare Register OCR0.

Timer/Counter Control Register TCCR0

7	6	5	4	3	2	1	0
FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00
0	1	1	0	1	0	1	1

Wir nutzen die Betriebsart “Fast PWM”. WGM01 = 1, WGM00 = 1.

Der Lüftertrainer erwartet positive Impulse. Also den Signalpegel am Anfang setzen und bei Zählwertgleicheit (Compare Match) löschen (Non-inverting Mode). COM01 = 1, COM00 = 0.

Also TCCR0 = 6BH.

Takterzeugung:
Der Zählerumlauf entspricht 256 heruntergeteilten Taktimpulsen. Wir brauchen eine Impulsfolgefrequenz von wenigen hundert Hz. 4 MHz : 256 = 15,6 kHz. Eine Taktteilung 1:64 ergibt rund 244 Hz. Also wie folgt probieren: CS2...0 = 3H.

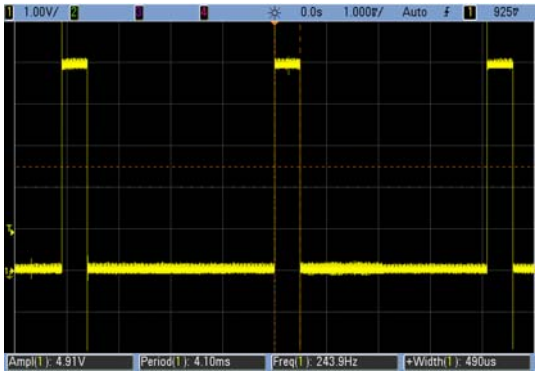
1. Die Zähler-Zeitgeber-Einheit 0 initialisieren.
Die LCD-Funktionen können im Quelltext gelöscht werden, die UART-Funktionen bleiben. Auf dem Oszilloskop muß eine näherungsweise symmetrische Impulsfolge zu sehen sein.

Programmbeispiel: 14.

Abb. 17 Die Impusfolge nach der Initialisierung.

2. Steuerung der Pulsweitenmodulation über die serielle Schnittstelle.
Wir werten die folgenden Zeichen aus:
+: Impulsbreite erhöhen (aufwärts) bis auf FFH.
–: Impulsbreite verringern (abwärts) bis auf Null.
0: Impulsbreite auf Null (Aus).
f: Impulsbreite auf FFH (Maximum, "Vollgas").

Programmbeispiel: 15.



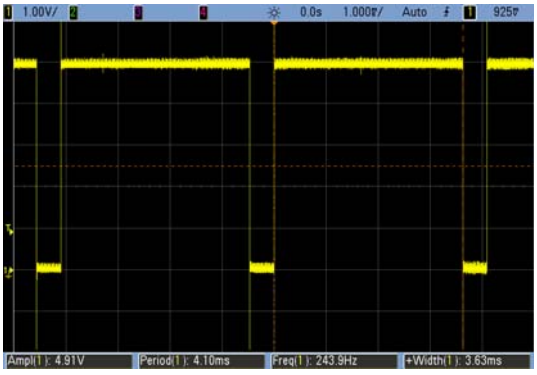


Abb. 18 Typische Impulsfolgen.

Wenn es auf dem Oszilloskop funktioniert, den Lüftertrainer anschließen.

Aufbau- und Inbetriebnahmevorschrift:
1. Stromversorgung (Festspannungsnetzgerät) aus.
3. Voltcraft-Netzgerät ein. Auf 12 V stellen. Wieder aus.
2. Lüftertrainer (Rückseite) an Voltcraft-Netzgerät anschließen. Schwarze Buchse an – (blau), blaue Buchse an + (rot).
3. Lüftertrainer an Port B des Starterkits anschließen.
4. Oszilloskop anschließen.
5. Oszilloskop ein.
6. Festspannungsnetzgerät ein.
7. Voltcraft ein.

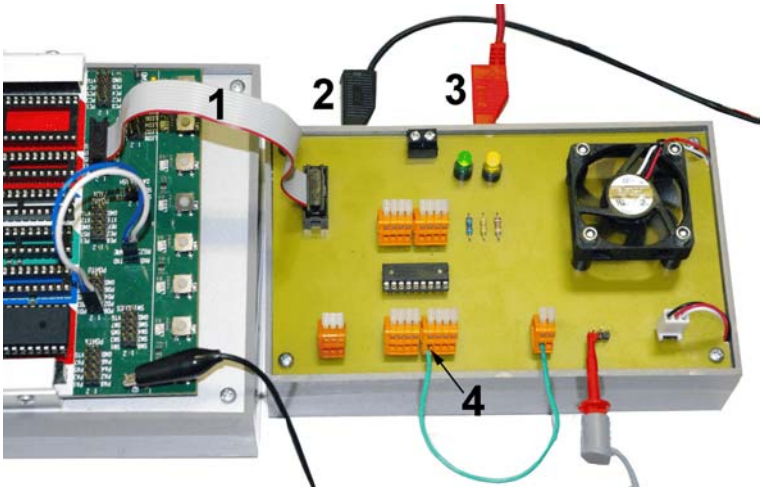


Abb. 19 Der Versuchsaufbau im Überblick. 1 - Port B. 2 - Masse Lüftermotor; 3 - Lüftermotor +12 V (2 und 3 mit Voltcraft-Netzgerät verbinden). 4 - Lüftermotorerregung über Bitposition 3. Ggf. das Brückenkabel entsprechend umklemmen.

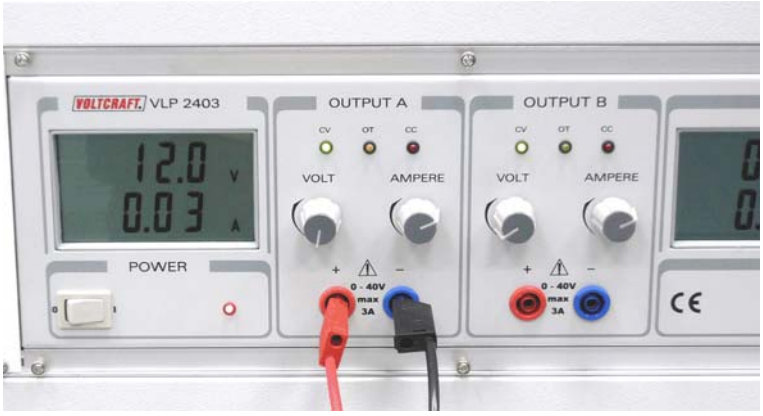


Abb. 20 So wird das Voltcraft-Netzgerät eingestellt und angeschlossen.

3. Den Bereich der Impulsbreite automatisch durchfahren (Wobbeln).

Die Impulsbreite und damit die Drehzahl des Lüfters soll von Null an langsam bis zum Maximalwert wachsen und dann ebenso langsam bis auf Null zurückgehen. Um diesen Ablauf ein- und auszuschalten, wird in der ISR zusätzlich das Zeichen “w” ausgewertet.

Programmbeispiel: 16.

Der ASCII-Code (Codetabelle)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20:		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30:	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40:	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50:	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60:	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70:	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

Beispiel: Zeichen “1” = 31H, Zeichen “A” = 41H.