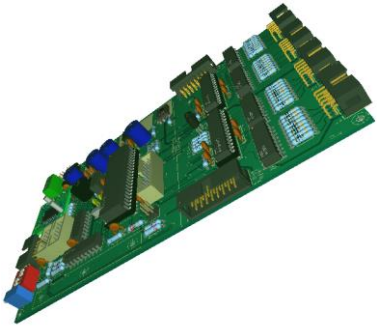
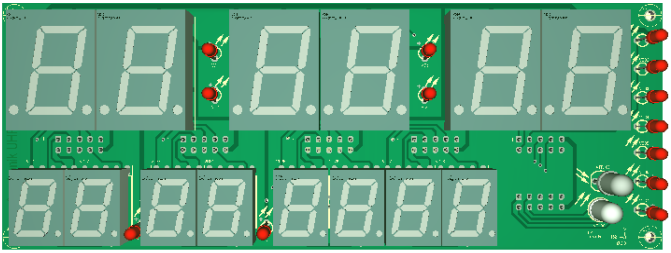
# - Handbuch -

# Digitalanzeige





Version : 1.81
Ausgabe : 2023-06
Autor : Ingolf Bauer
Kontakt : amatronik@arcor.de

# Inhaltsverzeichnis

**1 Allgemeines ..... 4**
1.1 Konzeptvorstellung ..... 4
1.2 Entwicklungssoftware ..... 4
1.3 Adresszuordnung ..... 5
1.4 Ansteuerung der Anzeigen ..... 6
**2 Programmierung (ISP) ..... 7**
2.1 Voraussetzungen und Treiberinstallation ..... 7
2.2 Einstellung der Fuse- und Lockbits ..... 17
2.3 EEprombereich des MC ..... 20
**3 Programmierung (HV) ..... 22**
3.1 Voraussetzungen und Vorgehensweise ..... 22
3.2 Software ..... 24
**4 Problemen beim Programmieren ..... 25**
4.1 Allgemeines ..... 25
4.2 Mikrocontroller (MC) wird nicht erkannt ..... 25
4.2.1 Kontakt- oder Treiberproblem ..... 25
4.2.2 Taktfrequenz ..... 25
4.2.3 Fusebits falsch eingestellt ..... 26
**5 Projekt Digitaluhr ..... 27**
5.1 Minimalversion für die Stunden- und Minutenanzeige ..... 27
5.1.1 Übertragung der Software ..... 27
5.1.2 Slave - Adressänderung (Firmware V 1.0.n) ..... 28
5.1.3 Slave - Adress-/ Anzeigeänderung (Firmware V 1.1.n) 29
5.1.4 Slave - Optionsänderung (Firmware V 1.2.n) ..... 29
5.1.5 Slave - 16-Segmentanzeigen (Firmware V 1.3.n) ..... 30
5.2 Ausbauversion ..... 31
5.3 Statusanzeige ..... 34
5.4 Bedienung ..... 35

5.4.1 Speicherposition der Einstellungen ..... 36

**6 Zusatzbaugruppen ..... 38**

6.1 DCF-Simulator mit Option Stromschleife ..... 38

**7 Ausführungen ..... 41**

7.1 Anzeige über Grafikdisplay ..... 41

7.2 Anzeige mit 14/16-Segment-Anzeigen..... 42

**8 Tools ..... 43**

8.1 Kodierung für 7-, 14- und 16-Segmentanzeigen..... 43

# 1 Allgemeines

## 1.1 Konzeptvorstellung

Für Digitalanzeigen, die nicht mit einem LCD- oder Grafikmodul realisiert werden sollen, ist diese Information vorgesehen.
Sie gibt eine Übersicht über die Gestaltungsmöglichkeiten digitaler Anzeigen mit Punktmatrix- oder 7-/14-/16-Segmentanzeigen, wobei gegenüber bisherigen Anzeigen, die in einem Multiplexbetrieb arbeiten, gemischte Bauteile (Typ oder Größe) zum Einsatz kommen können.

Ermöglicht wird dies durch den Einsatz einer Master- und einer individuell festlegbaren Anzahl an Slave-Baugruppen, solange der Adressbereich den I²C-Busses nicht überschritten wird.

Allerdings sei bereits zu Beginn darauf hingewiesen, dass dies bei diesem Konzept einen höheren Platzbedarf erfordert, zumal wegen einer einfacheren Bestückung auf Anwendung der SMD-Technik komplett verzichtet wurde, mit der eine wesentliche Platzreduzierung möglich wäre.
Mittlerweile sind eine Reihe von Versionen für den Aufbau verfügbar. Neben dem Einsatz von Grafikmodulen kann der Segmenttreiber HT16K33 eingesetzt werden, der einen platzsparenden Aufbau ermöglicht.

## 1.2 Entwicklungssoftware

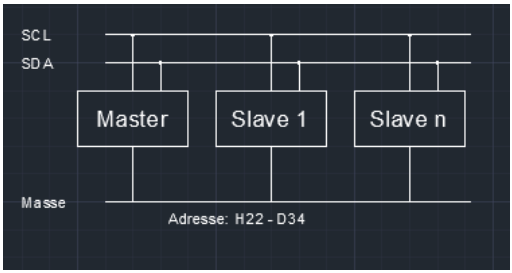
Die Firmware für Master und Slave ist Freeware; neben der kompilierten Version ist jeweils der Quelltext mit einer umfangreichen Kommentierung frei verfügbar.
Geschrieben wurde die Software unter der BASCOM-AVR IDE-Entwicklungsumgebung, einer interessanten und preiswerte Alternative vorhandenen professionellen Hochsprachen.
Der eingesetzte USBasp-Brenner ist darin direkt auswählbar, so dass mit einer kleinen Platine an einem USB-Port ein AVR problemlos beschrieben werden kann.

## 1.3 Adresszuordnung

An dieser Stelle soll nur kurz auf den Aufbau der Netzstruktur eingegangen werden, um einen Überblick zu erhalten. Weitergehende Informationen sind im Internet verfügbar.

Beim hier zur Anwendung kommenden I²C-Bus werden neben der Masseleitung die beiden Signalleitungen SCL und SDA benötigt, über die der Datenverkehr zwischen dem Master und den angeschlossenen Slaves stattfindet. Um eine eindeutige Zuordnung zu erhalten, muss jedem Slave eine individuelle Adresse zugeordnet sein.

Im Beispiel beginnt diese Adressierung mit 22 hexadezimal bzw. 34 dezimal.




Der Kern der zum Einsatz kommenden Baugruppen ist ein Mikrocontroller (MC), dem per Eingangsbelegung oder EEprom-Eintrag diese Adresse zugeordnet werden kann, was sich von Baugruppe zu Baugruppe unterscheiden kann und bei den Beispielen näher erläutert wird.

Bei der Adresszuordnung muss unbedingt darauf geachtet werden, welches System aktuell verwendet wird, da in der Praxis (z. B. Editoren für die Datei des EEproms) beide Systeme zur Anwendung kommen können, auf einer zweistelligen 7-Segmentanzeige jedoch nur die Darstellung im Hexformat möglich ist

## 1.4 Ansteuerung der Anzeigen

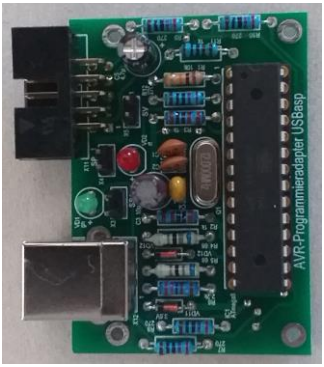
Die vorgestellten Schaltungen sind so ausgelegt, dass innerhalb bestimmter Grenzen jedes Anzeigeelement zur Anwendung kommen kann, dass - wenn es einen gemeinsamen Anschlusspunkt gibt - dieser die Anode ist.
Segmentanzeigen sind mit gemeinsamer Anode oder Katode verfügbar, aber Bauteile mit einer gemeinsamen Katode würden ein anderes Ansteuerkonzept bzw. eine Erweiterung der Anzeigetreiberstufen erfordern.
Durch die MC der Slaves werden folgende zwei Funktionen realisiert:

a) je zwei 8-kanalige Anzeige-Treiber sind über eine Adresse ansprechbar
b) was angezeigt werden soll, wird per Software festgelegt

Dieses Konzept ermöglicht es, unabhängig von Hardwaredekodern, alle mögliche Segmentvariationen, von Buchstaben über Ziffern bis zu Sonderzeichen alles anzeigen zu können.
Die Treiber sind für Segmentströme bis zu 500 mA ausgelegt, die für die meisten Anzeigen ausreichen sollten, da keine Stromspitzen auftreten, wie es bei Anzeigen im Multiplexbetrieb funktionsbedingt erforderlich ist.

Am Ende soll noch erwähnt werden, dass über dasselbe Konzept Relaiskarten betrieben oder I²C-Sensoren am selben Bus angeschlossen werden können, wenn man auf eine korrekte Adressvergabe achtet.
Daneben ist es möglich, über eine Soundausgabe Kuckucksrufe oder einen Stundenschlag erklingen zu lassen.

# 2 Programmierung (ISP)

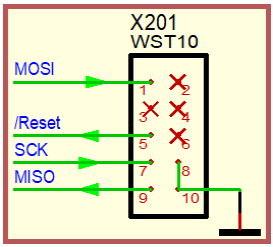



## 2.1 Voraussetzungen und Treiberinstallation

Bei den MC für Master und Slave handelt es sich moderne Bauelemente, die für die beschriebenen Funktionen mit einem Programm ausgestattet sein müssen.
Bei der Software handelt es sich um Firmware, die getestet und bereits mehrfach eingesetzt wurde.

Allerdings sind Änderungen in Bezug auf Funktionserweiterungen oder Optimierungen jederzeit möglich, so dass man überlegen sollte, sich selbst mit einem auf dem Markt angebotenen oder dem verfügbaren Programmiermodul in die Lage zu versetzen, die MC bei Bedarf zu aktualisieren.
Bei den Programmen zur Steuerung und Anzeige handelt es sich um Software, deren Quelltext frei zugänglich und umfangreich kommentiert ist.
So sind in ihr beispielsweise die Einstellungen für die Fuse- und Lockbits hinterlegt.

Was wird benötigt?
Zunächst ein PC oder Notebook mit einer freien USB-Anschlussmöglichkeit für das seriell arbeitende Programmiermodul. Wenn das Programmiermodul mit einer ISP-Schnittstelle (10-poliger Wannensteckverbinder) ausgestattet ist, kann der MC zum Programmieren in der Schaltung belassen werde, da die Basisplatine über einen entsprechenden Anschluss verfügt.



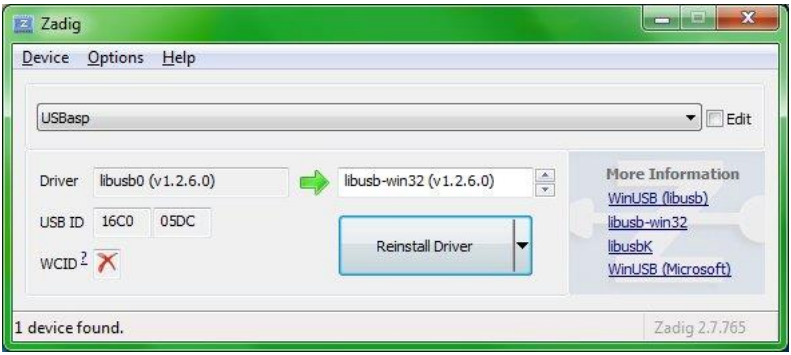

*Allgemeine PIN-Belegung des Programmieranschlusses*

Über eine serielle Schnittstelle (SPI) erfolgt dann der Zugriff auf die Bereiche
- Fuse- und Lockbits
- EEprom
- Flash (Programmspeicher)

Da ich bereits gute Erfahrungen beim Einsatz des USBasp-Moduls (https://www.fischl.de/usbasp) gemacht habe, beziehen sich alle weiteren Hinweise auf dieses Programmiermodul.
Entweder man erwirbt ein fertiges Modul oder nutzt die von mir angebotene Leiterplatte für einen Aufbau. Allerdings möchte ich darauf hinweisen, dass für die Funktion ein bereits programmierter MC benötigt wird.

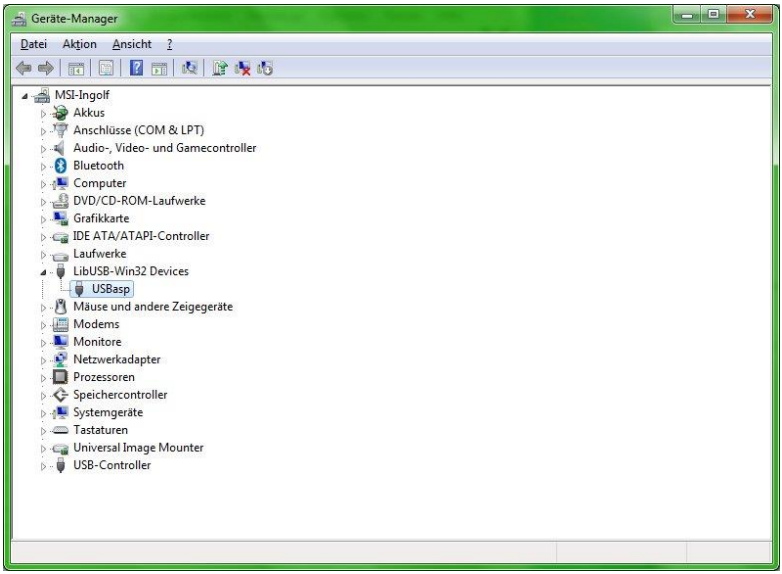
Wird das Modul über ein USB-Kabel mit dem Computer verbunden, muss zunächst ein Treiber installiert werden. Es können dazu im Internet eine Reihe von Anleitungen nachgelesen werden.
Die Vorgehensweisen ähneln sich und kurz zusammengefasst beinhalten sie folgenden Schritte:

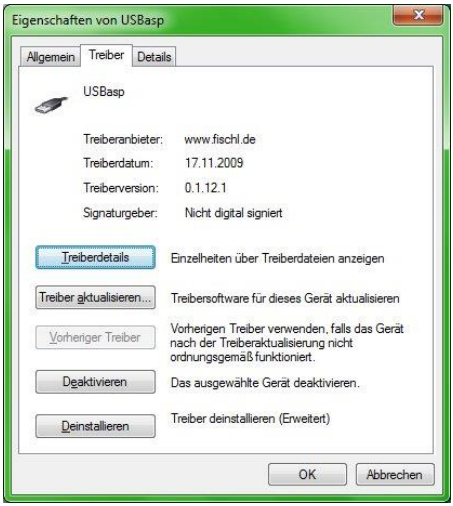
- nach der Software Zadig V2.7 googeln, herunterladen und installieren (https://zadig.akeo.ie, Stand 01.11.2021)
- Zadig starten, über das Menü Gerät und den Treiber auswählen
  → Options
  → „List all Devices“ anklicken
  → „USBasp“ auswählen
  → Treiber libusb ( v1.2.6.0) auswählen
  → „Install Driver“ anklicken




*Menüoberfläche des Programms*

Danach ist bei einer korrekten Installation im Gerätemanager das Gerät „USBasp“ ohne gelbe Markierung eingetragen.

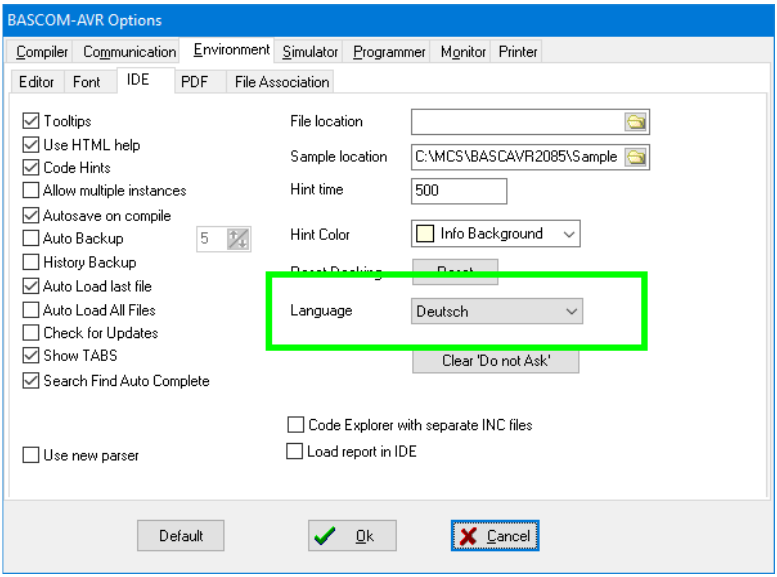





Um den MC zu konfigurieren, kann die Entwicklungsumgebung Bascom eingesetzt werden.

Der Link
https://www.mcselec.com/index.php?option=com_docman&task=cat_view&gid=73&Itemid=54
führt über „BASCOM-AVR“ zur Demoversion „BASCOM-AVR Demo“, die auf einen Programmcode von 4 kB begrenzt ist, was aber nicht weiter stört, da wir kein Programm entwickeln wollen, sondern eine fertig kompilierte Datei übertragen möchten.

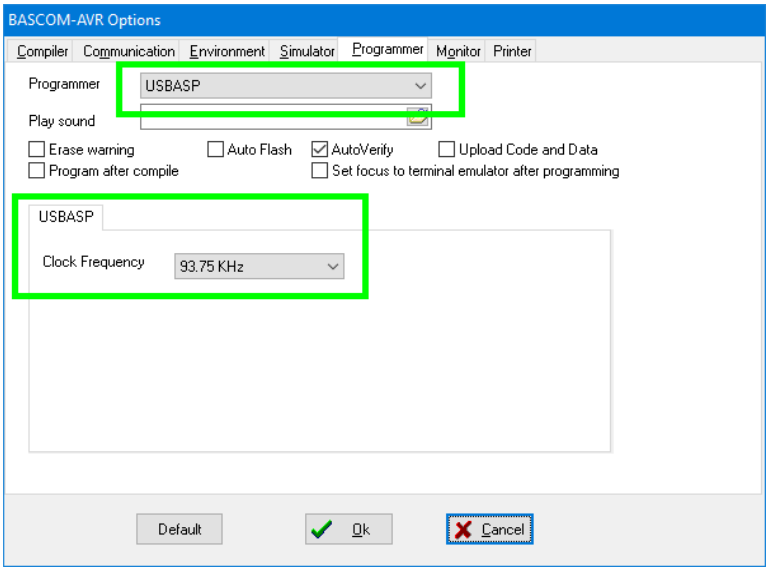
Nach der Installation des Programms sollte zunächst auf die deutsche Sprachführung umgestellt werden:




*Spracheinstellung*

Leider sind nicht alle Befehle und Hinweise übersetzt worden; meist hilft dann der Google Übersetzer weiter.

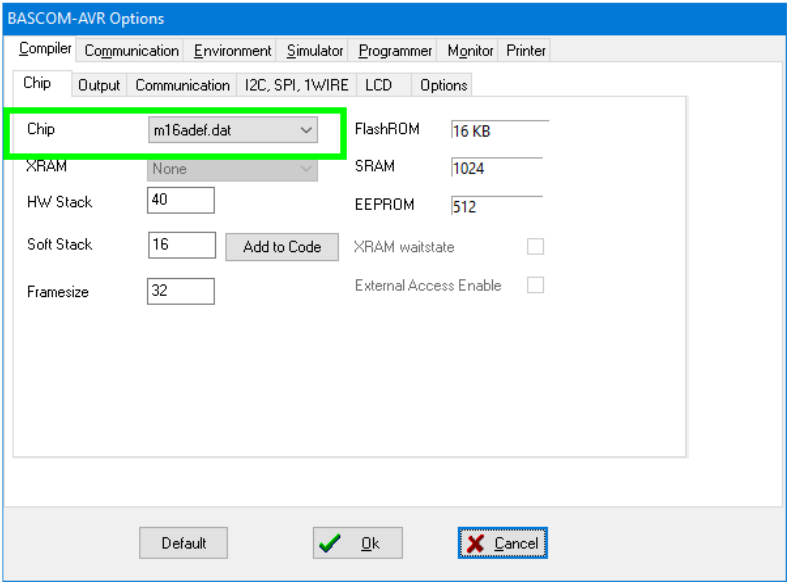
Im Registerblatt „Programmer“ ist das Gerät „USBasp“ auszuwählen. Im Optionsfeld können eine Reihe von Einstellungen vorgenommen werden, auf die ich hier nicht näher eingehen möchte.




*Gerät USBasp und Taktfrequenz wählen*

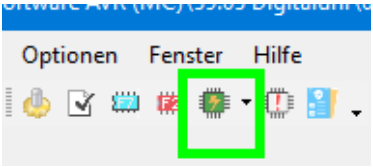
Wenn eine Kommunikationsstörung zwischen Bascom und dem angeschlossen AVR vorliegt, kann es daran liegen, dass dieser Wert zu hoch oder ein falscher AVR-Typ im Register „Compiler“ und dort „Chip“ eingestellt ist.
Die dort eingetragene Definitionsdatei (*.dat“) muss mit dem angeschlossenen AVR-Typ identisch sein. Dabei ist z. B. auch zwischen einem ATmega8 und einem ATmega8A zu unterscheiden.

Hat man hier einen zu kleinen Wert eingetragen, dauert bei großen Datenmengen die Übertragung sehr lange.




*Festlegung der Definitionsdatei*

Über die Schaltfläche für das Programmieren gelang man zur Oberfläche des „USBASP Programmers“, in der man über die Registerblätter Zugriff auf den Flash, das EEprom sowie die Lock und Fusebits hat.




*Icon zum Programmieren (F4)*

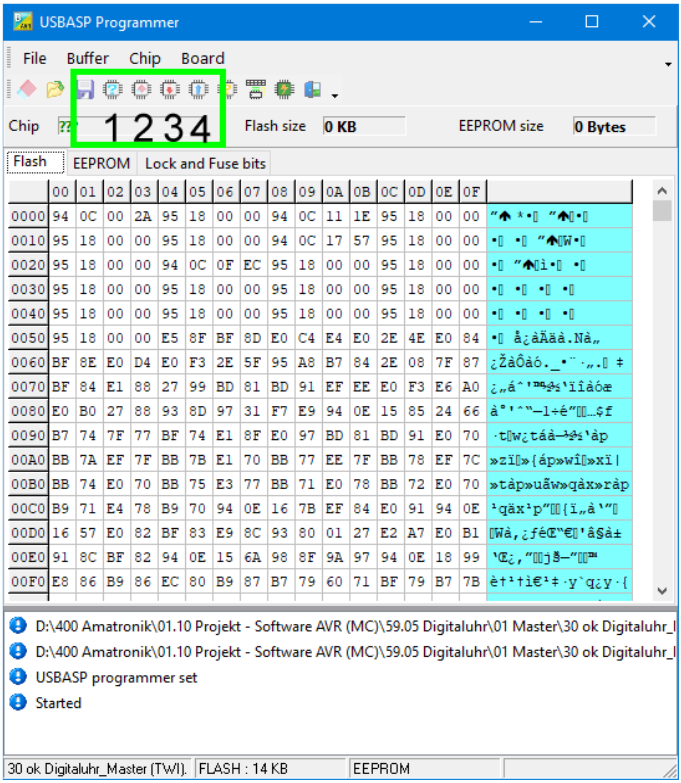
Bedeutung der Schaltflächen:

1 Identifizierung des AVR-Typs
(Identify chip)

2 AVR löschen
(Erase chip))

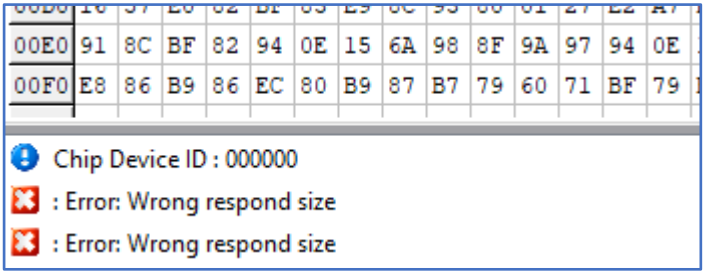
3 Daten des Zwischenspeichers übertragen
(Write buffert to chip)

4 AVR-Daten in den Zwischenspeicher übertragen
(Read code from chip into buffer)

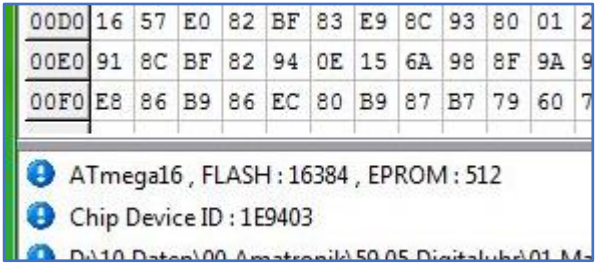



*Bedienoberfläche des Programmers*

Zunächst sollte getestet werden, ob eine Kommunikation zum MC möglich ist.
Über die Schaltfläche 1 (Identify chip) wird auf den MC zugegriffen und im unteren Fensterbereich erscheint das Ergebnis der Prüfung:




*Beispiel eines fehlgeschlagenen Verbindungsversuches*

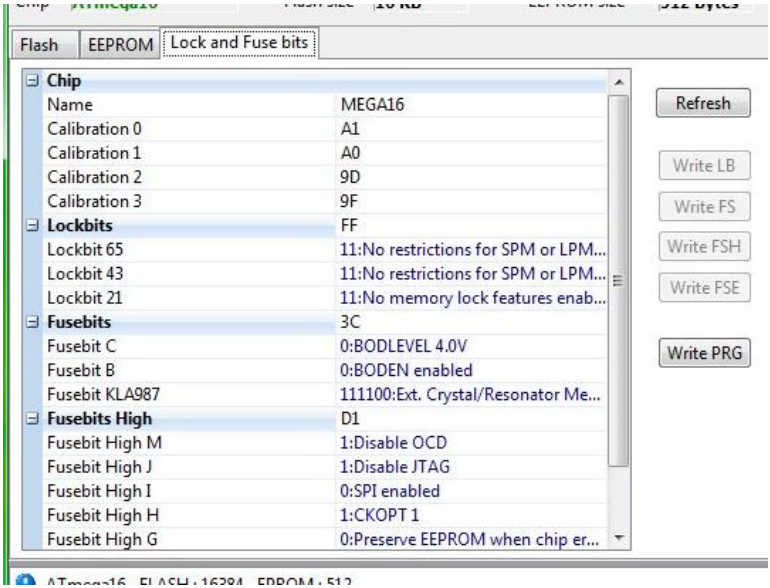



*Beispiel einer korrekten Verbindung zu einem ATmega16*

## 2.2 Einstellung der Fuse- und Lockbits

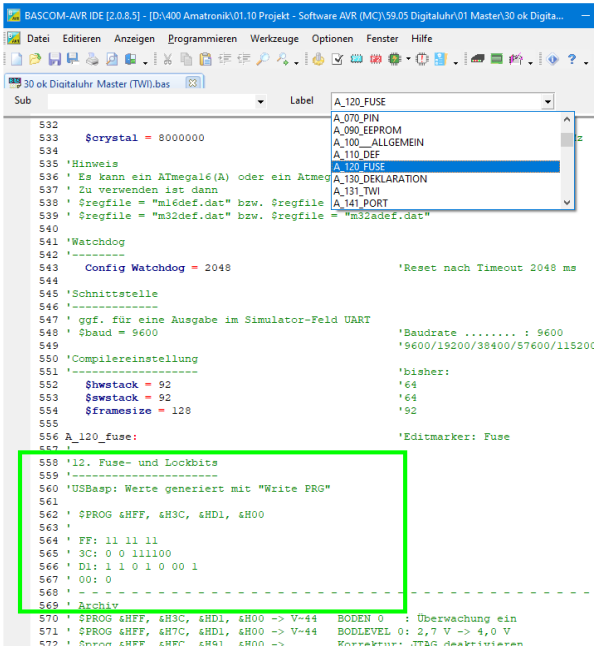
Nach dem Kommunikationstest ist zu prüfen, ob die Fuse und Lockbits korrekt eingestellt sind.
Die Einstellungen im Auslieferungszustand entsprechen meist nicht den vom Programmierer gewählten Optionen. Daher ist der erste Blick auf diesen Bereich von großer Bedeutung.

Dazu wählt man das Registerblatt „Lock und Fuse bits“ und vergleicht die Einstellungen mit den Angaben im Quelltext des Programms.
Bei einer abweichender Einstellung kann am Ende der entsprechenden Zeile über ein sich aufklappendes Menü die korrekte Biteinstellung ausgewählt werden.
Danach erscheint die entsprechende Schaltfläche rechts im Bild nicht mehr ausgegraut und mit einem Klick werden die neuen Werte im MC verändert.



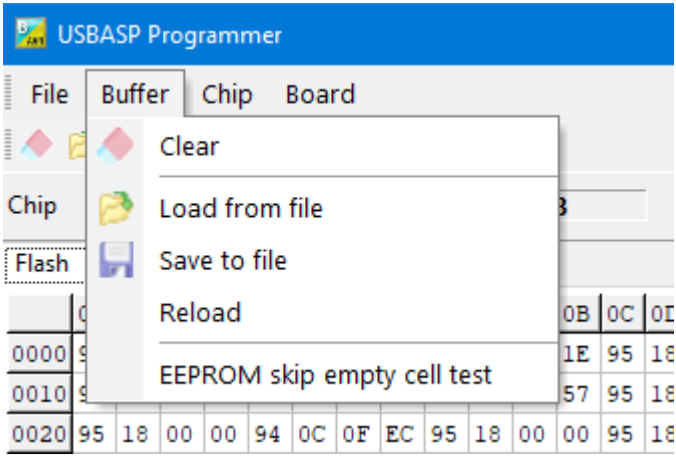

*Übereinstimmung der Einstellungen mit dem Quelltext (3C, D1)*

Hinter jeder Einstellung verbirgt sich eine bestimmte Funktion, die in den Datenblättern nachgelesen werden kann. Sie sind aber eher nur dann von Bedeutung, wenn man selbst ein Programm schreiben möchte.
Beispielsweise muss man festlegen, ob ein interner Oszillator für den Systemtakt zuständig sein soll oder ein externer Quarz mittlerer Frequenz dafür vorgesehen ist. Möglich ist weiterhin der Anschluss eines externen Taktgebers. Dementsprechend sind dann die Bits einzustellen.
Ich habe es mir zur Gewohnheit gemacht, die Werte stets an dieser Stelle einzutragen. Über das Auswahlmenü gelang man schneller zu diesem Bereich, da darüber der Marker „A_120_fuse“ direkt angesprungen werden kann.
Im vorgestellten Projekt stellen sich die Fuse- und Lockbits wie folgt dar:


BASCOM-AVR IDE [2.0.8.5] - [D:\400 Amatronik\01.10 Projekt - Software AVR (MC)\59.05 Digitaluhr\01 Master\30 ok Digita...
Datei Editieren Anzeigen Programmieren Werkzeuge Optionen Fenster Hilfe
30 ok Digitaluhr Master (TWI).bas
Sub
Label
A_120_FUSE
A_070_PIN
A_090_EEPROM
A_100__ALLGEMEIN
A_110_DEF
A_120_FUSE
A_130_DEKLARATION
A_131_TWI
A_141_PORT
532
533 $crystal = 8000000
534
535 'Hinweis
536 ' Es kann ein ATmega16(A) oder ein Atmeg
537 ' Zu verwenden ist dann
538 ' $regfile = "m16def.dat" bzw. $regfile
539 ' $regfile = "m32def.dat" bzw. $regfile = "m32adef.dat"
540
541 'Watchdog
542 '--------
543 Config Watchdog = 2048 'Reset nach Timeout 2048 ms
544
545 'Schnittstelle
546 '-------------
547 ' ggf. für eine Ausgabe im Simulator-Feld UART
548 ' $baud = 9600 'Baudrate ..... : 9600
549 '9600/19200/38400/57600/115200
550 'Compilereinstellung
551 '-------------------- 'bisher:
552 $hwstack = 92 '64
553 $swstack = 92 '64
554 $framesize = 128 '92
555
556 A_120_fuse: 'Editmarker: Fuse
557
558 '12. Fuse- und Lockbits
559 '----------------------
560 'USBasp: Werte generiert mit "Write PRG"
561
562 ' $PROG &HFF, &H3C, &HD1, &H00
563 '
564 ' FF: 11 11 11
565 ' 3C: 0 0 111100
566 ' D1: 1 1 0 1 0 00 1
567 ' 00: 0
568 ' - - -

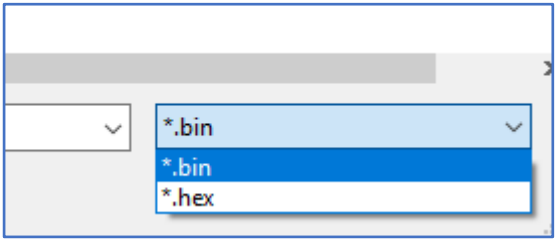
## Programmcode übertragen

Kommen wir zum nächsten Schritt, dem Übertragen des Programmcodes.
Über das Menü „Buffer“ kann mit dem Befehl „Load from file“ das entsprechende Hex-File in den Zwischenspeicher geladen werden.




*Daten in den Zwischenspeicher laden*

Dabei ist darauf zu achten, dass die Dateiendung „.hex“ gewählt wird.



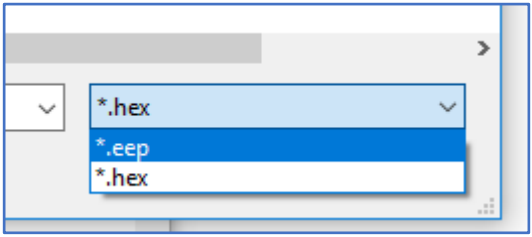

*Dateiformat „*.hex“ ist zu wählen*

Nach der Dateiauswahl wird über sie Schaltfläche 3 (Write buffert to chip) der Programmcode an den MC übertragen. Rechts unten kann diese Übertragung an einem Fortschrittsbalken verfolgt werden.

Damit ist der MC „gebrannt“ und wenn alles korrekt abgelaufen ist, startet die Programmabarbeitung.
Sollte dies ausbleiben, kann zunächst über die Schaltfläche 2 der Programmspeicher den MC gelöscht und neu beschrieben werden. Oft ist der Fehler auch behoben, wenn man die Stromversorgung unterbricht und wieder neu einschaltet.

## 2.3 EEprombereich des MC

Der MC verfügt neben dem Programmspeicher auch über einen EEprom-Bereich, dessen Programmierung ähnlich erfolgt.
Es ist nur darauf zu achten, dass dazu das Register „EEPROM“ gewählt und auf die Dateiendung „.eep“ geachtet wird, wenn die Daten in diesem Format bereitgestellt werden.




*Dateiformat „*.eep“*

Dieser EEprom-Bereich wird bei dem hier vorgestellten Projekt dafür genutzt, bestimmte Einstellungen (Grundhelligkeit, Slaveadresse, …) vornehmen zu können, die nach einem Neustart wieder zur Verfügung stehen sollen.
Programmtechnisch erfolgt das über eine bestimmten Deklaration dieser Variablen.
Man kann die Programmieroberfläche darüber hinaus gezielt dafür nutzen, diese Werte zu verändern.
Wie sicherlich noch in Erinnerung ist, muss im Netzwerk jedem Slave eine eindeutige Adresse zugeordnet sein. Dies kann während des Betriebes über das Pin 10 (Kanal B.7) des ATmega8A erfolgen, oder aber bereits bei nach der Programmierung durch Beschreiben der EEprom-

Adresse &H01.
Bei der Datenansicht von „Flash“ oder „EEPROM“ kann man zwischen der dezimalen oder hexadezimalen Darstellung wählen.
Bei der Wertezuweisung zu den Speicherzellen muss dabei stets darauf geachtet werden, dass die Eingabe entsprechend dem dargestellten Zahlensystem entspricht.
Beispielsweise entspricht der Wert &H22 dezimal 34 und &H3C dezimal 60! Eine Hilfestellung kann dabei der in Windows verfügbare Rechner geben, mit der eine Umrechnung vorgenommen werden kann.

Zum Anfang hört sich alles ein wenig kompliziert an, aber wenn man sich damit beschäftigt, fällt es immer leichter, die MC zu programmieren.

An dieser Stelle möchte ich auf mein kostenfreies Angebot hinweisen, bei Bedarf online eine kleine Einführung zu geben.
Mit dem TeamViewer (für den Privatgebrauch freier Download unter https://www.teamviewer.com/de/download/windows) ist es mir möglich, bei Ihrer Einwilligung per Passwort über einen Fernzugriff (Internet) die oben erläuterten Verfahren am PC vorzustellen.

Anfragen und Hinweisen bitte an amatronik@arcor.de

# 3 Programmierung (HV)

## 3.1 Voraussetzungen und Vorgehensweise

Ergänzend zu der „normalen“ Programmierung möchte ich in diesem Abschnitt auf die sogenannte Hochspannung-Programmierung eingehen.
Über diese Schnittstelle ist es auch dann noch möglich, sich mit dem MC zu verbinden, wenn man beispielsweise durch ein falsches Setzen der Fuse-Bits sich selbst „ausgesperrt“ hat.

Dazu wird am Pin für das Reset-Signal eine Spannung von +12 V angelegt. Aber dies ist nicht ausreichend, sondern es muss aus dem Datenblatt entnommen werden, ob sich der jeweilige MC für eine serielle (HVSP) oder parallele (PP) Programmierung eignet. Da für diese Art der Programmierung der MC anders zu beschalten ist, kann nur mit einer dafür geeigneten Hardware gearbeitet werden. Nur in den wenigsten Fällen ist eine In-System-Programmierung möglich, da die Schaltung für die 12V-Spannung am Reset-Pin ausgelegt sein muss.

Beispielsweise kann ein STK500 für diese Art der Programmierung eingesetzt werden. Unter dem Link

https://www.mikrocontroller.net/articles/AVR_HV-Programmer sind weitere HV-Programmer aufgeführt.

Da in diesem Projekt die MC ATmega8A und ATmega16A verwendet werden und ich bei einer Recherche auf einen HV-Brenner gestoßen bin, der als Basis einen Arduino 2560 R3, ein Shield als Erweiterung verwendet und genau diese Typen unterstützt, möchte ich dieses „open-source hardware project“, was unter Github (https://github.com/dilshan/avr-hv2) verfügbar ist, kurz vorstellen.

Was wird benötigt?
Zunächst ein Arduino ATmega 2560 R3, wobei ich eine kostengünstige Variante eingesetzt habe, bei der als USB-Schnittstelle kein AVR, sondern der CH340 einsetzt wird.
Der Arduino verfügt über eine USB-Schnittstelle, die mit einem PC verbunden wird. Weiterhin ist eine 12V-Stromversorgung erforderlich, die für die Programmierspannung erforderlich ist.
Weiterhin wird eine Leiterplatte benötigt, um die Hardwareerweiterung (shield) bestücken zu können. Sie kann über den Hersteller „PCBWay“ bezogen werden; einige Platinen habe ich noch vorrätig. Aber es gibt bei dieser Version folgende Nachteile, auf die ich beim Aufbau gestoßen bin:

1. Der Entwickler hat Transistoren japanischer Bauart verwendet, die sich gegenüber den bei uns eingesetzten in der Reihenfolge der Anschlüsse unterscheiden.
Dieses Problem lässt sich leicht beim Bestücken durch das Kreuzen der Anschlüsse korrigieren.

2. Möchte man auch für den 28-poligen Anschluss einen Textoolsockel einsetzen, muss man bei der Bestellung sehr genau hinsehen, da die Leiterplatte für einen Reihenabstand von 10,16 mm ausgelegt ist; typisch sind allerdings 7,62 mm.
Da es mir nach ausgedehnten Recherchen nicht möglich war, einen preisgünstigen Sockel zu erwerben, habe ich mich entschlossen, eine Leiterplatte als Zwischenadapter zu entwickeln, die ebenfalls noch mit einigen Exemplaren verfügbar ist.

3. Die Anwendersoftware ist unter Windows und Linux einsetzbar,

allerdings nur in einer 64-bit-Umgebung. Ältere Notebook können daher unter Umständen nicht verwendet werden.

**Hinweis:**
Ab 06/22 ist eine überarbeitete Leiterplatte verfügbar, bei der die Anschlüsse für die Transistoren und den Textoolsockel verändert sind.

## 3.2 Software

Um den Brenner benutzen zu können, möchte ich die einzelnen Schritte kurz aufführen:

**- Treiber für die USB-Schnittstelle**
Für den Einsatz der im nächsten Schritt aufgeführten Installation der Arduino-Entwicklungsumgebung ist eine Kommunikation mit einem Computer erforderlich.
Sie bringt zwar einige Treiber mit, aber wenn man einen Arduino mit dem CH340 besitzt, muss man dies in Eigenregie anpassen. Dazu kann nach dem Entpacken der entsprechenden ZIP-Datei dieser installiert werden.

**- Arduino programmieren**
Dafür ist die kostenfreie IDE für den Arduino einzusetzen. Die aktuelle Version 2.0 macht zwar optisch den besseren Eindruck, aber da bei mir die Spracheinstellung nicht einstellbar war, wurde die Version 1.8.19 installiert.
Zunächst ist der korrekte Port und Typ der angeschlossenen Hardware auszuwählen.
Innerhalb der Projektdaten ist eine Datei mit der Endung *.ino verfügbar, die Sketchdatei „hvprog2-firmware.ino“.
Diese ist per Upload auf den Arduino zu schreiben.

**- Anwendersoftware**
Unter Windows kommt für die Funktionen des Brenners das Programm „arduhvprog2.exe“ zum Einsatz, über das man Zugriff auf die Fuse- und Lockbits sowie den Flash- und EEprom-Speicher erhält.

# 4 Problemen beim Programmieren

## 4.1 Allgemeines

Die in den folgenden Punkten beschriebenen Probleme ist eine Auflistung von möglichen Ursachen für das Fehlschlagen der Programmierung, die auf eigenen Erfahrungen beruhen.
Weitergehende Hinweise zur Thematik finden sich im Internet.

## 4.2 Mikrocontroller (MC) wird nicht erkannt

### 4.2.1 Kontakt- oder Treiberproblem

**Symptom:**
Der MC kann nicht auf eine neue Firmwareversion umgestellt werden.

**Ursache:**
Treiber- oder Kontaktproblem

**Fehlerbehebung:**
Zunächst ist zu prüfen, ob die Anschlüsse für die Stromversorgung und der ISP-Schnittstelle (Signale MISO, MOSI, SCK, Reset) fehlerfrei sind.
Im Windows Gerätemanager ist die fehlerfreie Installation des Gerätetreibers für das Programmiergerät zu überprüfen.
Abhilfe kann in Einzelfällen durch den Einsatz eines anderen USB-Ports geschaffen werden.

### 4.2.2 Taktfrequenz

**Symptom:**
Die Kommunikation zum MC ist gestört.

**Ursache:**
Die eingestellte Frequenz für das USBasp-Modul passt nicht zur Taktfrequenz des MC.

**Fehlerbehebung:**
Unter Bascom kann im Optionsmenü für das USBasp-Modul eine Taktfrequenz eingestellt werden.
Ist sie sehr klein gewählt, dauert es bei größeren Firmware-Updates

ggf. sehr lange, bis der Brennvorgang abgeschlossen ist.
Bei zu hoch gewählten Einstellungen ist der MC nicht ansprechbar.

### 4.2.3 Fusebits falsch eingestellt

**Symptom:**
Fabrikneue MC lassen sich mit dem USBasp-Modul nicht ansprechen

**Ursache:**
Der Grund dafür war, dass die Fusebits bei Auslieferung auf den Anschluss eines Quarzes als Taktgeber eingestellt waren und damit in einer Schaltung, die für den Einsatz des internen RC-Oszillators vorgesehen war, nicht funktionieren konnten.

**Fehlerbehebung:**
Nachdem die MC korrekt mit einem Quarz beschaltet waren (Experimentierboard), konnten die Fusebits umgestellt werden.
Sie ließen sich danach in der vorgesehenen Schaltung wieder problemlos beschreiben und auslesen.

# 5 Projekt Digitaluhr

## 5.1 Minimalversion für die Stunden- und Minutenanzeige

Für diese Projekt wurde eine Masterbaugruppe entwickelt, auf der bereits zwei Slaves angesteuert werden können.
Alleine mit dieser Baugruppe ist es daher möglich, eine einfache Digitaluhr mit dem Anzeigeformat „hh:mm“ zu realisieren.
An jedem Slave sind zwei 8-kanalige Treiber angeschlossen, die je einem Digit der Anzeige zugeordnet sind.
Für die Zifferndarstellung sind 7 Kanäle zuständig und beim achten kann man sich entscheiden, den auf den Anzeigen befindlichen Punkt zu nutzen oder separate LED als Trennzeichen zu verwenden. Die Einstellung erfolgt per Lötpads auf der Leiterplatte für die Anzeige.

### 5.1.1 Übertragung der Software

Da es sich beim Master, wie auch bei den Slaves um Mikrocontroller handelt, ist es zunächst erforderlich, sie mit der jeweiligen Firmware auszustatten.
Dazu ist auf der Masterbaugruppe ein Programmierstecker für den Anschluss an einen Brenner vorhanden. Beim Brenner handelt es sich um eine Variante des USBasp, der direkt über das Bascom-SDK angesprochen werden kann.
Aber es ist möglich, bereits vorbereitete MC oder andere Brenner einzusetzen.

Alle über die Netzstruktur verbundene Baugruppen sind parallel an die für das Brennen erforderlichen Signalleitungen angeschlossen. Nur das Reset-Signal kann per DIP-Schalter jedem MC einzeln zugeschaltet werden.

Achtung!
Bei einem Brennvorgang darf nur ein DIP-Schalter geschlossen sein.

Handelt es sich um bisher ungenutzte MC, sind zunächst die Fuse- und Lockbits einzustellen.
Hinweise dazu sind der jeweiligen Firmwareversion zu entnehmen.

Für die Funktion als Digitaluhr sind für Master- und Slave getrennte Firmware-Versionen einzusetzen. Die Masterversion ist z. Z. nur für den Betrieb einer vollen Zeit- und Datumsanzeige verfügbar, kann aber für Minimalversionen schnell angepasst werden.
Manuelle Einstellungen ermöglichen es im Betrieb, die erforderlichen Änderungen (z. B. Anpassung der Slave-Adresse) durchzuführen.

Es ist aber auch möglich, für jede Baugruppe eine zugeschnittene Datei für das EEprom auf dem MC abzulegen, da optionale Einstellungen, z. B. für die Grundhelligkeit der Anzeige oder der Slave-Adresse, auf bestimmten Speicherplätzen hinterlegt sind.

Da das Signal „Reset“ bei jedem Brennvorgang an einem Kanal des Masters (PD.3, Pin 17) zur Verfügung steht und in der Software abgefragt wird, erfolgt automatisch nach jeder Programmierung ein Neustart, was ansonsten manuell ausgeführt werden sollte.
Die Notwendigkeit für einen Neustart ergibt sich daraus, dass bei Änderungen der Busstruktur eine neue Synchronisation stattfinden sollte.

### 5.1.2 Slave - Adressänderung (Firmware V 1.0.n)

Der MC für den Betrieb als Slave gewählte Atmega8A verfügt über einen Kanal (PB.7, Pin 10), der nicht für die Ansteuerung der Anzeige benötigt wird.
Legt man diesen über den DIP-Schalter auf Massepotential, wird dies von der Software erkannt und in der Anzeige (Adresse H22) erscheint „SL“ für „Slave“.
Nach einer kurzen Zeit wechselt die Anzeige in eine Zifferndarstellung, über die die eingestellte Slave-Adresse ablesbar ist.
Wird weiterhin das Pin auf Massepotential gehalten, wird die Adresse in Zweierschritten hochgezählt, wobei der Bereich so gewählt ist, dass die Adressierung für eine voll ausgebaute Zeit- und Datumsanzeige möglich ist.
Bei der hier vorgestellten Minimalversion sind nur die dezimalen Adresse 34 für die Stunden- und 36 für die Minutenanzeige relevant.

### 5.1.3 Slave - Adress-/ Anzeigeänderung (Firmware V 1.1.n)

Die Firmware nach dem vorigen Abschnitt wurde um die Option zur Auswahl der Wochentag-Anzeige erweitert (V 1.1.n).
Die Auswahl besteht zwischen einer Punktanzeige (je Tag einen LED) oder einer Bandanzeige, bei der die LED-Anzahl die vergangenen Tage sowie dem aktuellen entspricht.
Diese Auswahl kann bereits bei der Programmierung des MC getroffen werden, indem auf die Adresse &H02 der Wert 0 bzw. 1 geschrieben wird. Eine „0“ steht für eine Punktanzeige.

Alternativ kann die Auswahl über das bereits erwähnte Pin 10 (PB.7) erfolgen.
Dazu muss **bereits vor dem Einschalten der Betriebsspannung** das Pin mit Masse verbunden (Schalter) und nach ca. 1 Sekunde unterbrochen werden, da ansonsten das MC-Programm nicht startet und der Slave nicht erkannt wird.

### 5.1.4 Slave - Optionsänderung (Firmware V 1.2.n)

Neben der im Punkt 2.1.2 aufgezeigten Möglichkeit der Adressänderung ist eine Software für die Slaves in der Entwicklung, die weitere Optionen bereitstellt.
Im Normalbetrieb werden die Programmiereingänge nicht benötigt. Dies ermöglicht es, zusätzlich zu einer Adressänderung weitere Einstellungen vornehmen zu können, wenn diese Eingänge mit Massepotential belegt werden.
Es ist z. B. angedacht, unabhängig von der Helligkeitsregelung durch den Master jedes Digit bzw. die Trennanzeige in ihrer Helligkeit per PWM veränderbar zu machen.
Da es sich allerdings bei Tests herausstellte, dass noch nicht weiter untersuchte Ursachen dazu führen, dass die Anzeigen sichtbar zu flackern anfangen, befindet sich diese Firmware noch im Entwicklungsstadium. Wahrscheinlich sind Interferenzen der überlagerten PWM der Grund für diese Erscheinung.

Ein bedeutender Vorteil bei dieser Version wäre es, relativ unabhängig von der einmal getroffenen Festlegung der Segmentwiderstände zu sein, so dass weiter nach einer Problemlösung gesucht wird.

Es könnte z. B. daran liegen, dass sich zwischen der Frequenz der PWM, die durch den Master vorgegeben wird und der angedachten PWM der Steuerung der einzelnen Digits ein Schwebezustand einstellt, der das Flackern bewirkt. In diesem Fall würde die Lösung in der Abschaltung der Master-PWM liegen.

Da der Helligkeitssensor nur vom Master ausgewertet wird, muss die Information einer „Grundhelligkeit“ an alle Slaves über einen anderen Weg erfolgen.
Denkbar wäre es, dafür den Datenbus zu verwenden, was allerdings eine Änderung der Master-Firmware zur Folge hätte.
Und wenn man schon ein Datenpaket für die Slaves überträgt, könnte dies auch andere Informationen beinhalten, was eine Steuerung über die Programmiereingänge hinfällig machen würde.

### 5.1.5 Slave - 16-Segmentanzeigen (Firmware V 1.3.n)

Die Slave-Erweiterungsbaugruppe aus dieser Versionsreihe ist dafür ausgelegt, 16-Segment-Anzeigen anzusteuern.

*Anzeige PSA08-11 und VQB201 (DDR)*

Damit lassen sich optisch ansprechendere Darstellungen - beispielsweise für die Wochentagsanzeige - realisieren.

Für die Montage der Elemente kann dieselbe Leiterplatte verwendet werden. Die Hersteller haben allerdings eine voneinander abweichende Zuordnung der Segmente zu den Anschlusspins vorgesehen, so dass für die Anzeige eine andere Kodier-Tabelle eingesetzt werden muss.
Für beide Anzeigen ist diese Tabelle in der Software hinterlegt. Die Steuerung der Auswahl erfolgt über die EEprom-Variable &H03.

Bei dem Wert “0“, der automatisch beim ersten Einschalten in das EEprom geschrieben wird, wenn sich noch keinerlei Daten darin befinden, wird die Kodier-Tabelle für die Anzeigen aus der Reihe PSA08-11 und deren pinkompatiblen Versionen verwendet.
Der Typ VQB201, der als DDR-Typ teilweise sehr kostengünstig zu erwerben ist, kann korrekt angesprochen werden, wenn der Wert des EEproms auf der Adresse &H03 auf „1“ geändert wird.

Zu erwähnen ist, dass jede andere Anzeige mit gemeinsamer Anode anschließbar ist. Für eine korrekte Anzeige ist ggf. eine modifizierte Kodier-Tabelle in der Software zu hinterlegen.

## 5.2 Ausbauversion

Je nach Ausführung der Firmware für den Steuerprozessor können verschiedene Digitalanzeigen realisiert werden.

Nach dem Einschalten der Baugruppe wird zunächst geprüft, ob die Anzahl bzw. Adressierung der am Bus vorhandenen Slaves mit der in der Software eingestellten Anzahl übereinstimmt.
Jede korrekte Adressierung wird durch einen kurzen Signalton quittiert. Tritt bei der Kontrolladressierung ein Fehler auf (langer Ton), bedeutet das, dass der Slave mit der erforderlichen Adresse nicht identifiziert werden konnte, was meist auf eine falsche oder doppelte Adresszuordnung hindeutet.
Bei größeren Buslängen sollte ein aktiver Busabschluss zum Einsatz kommen, für dessen Realisierung die Treiberplatinen bereits vorbereitet sind.

Der Wert für die Anzahl der Slaves ist im EEprom abgelegt und kann über die Menüführung verändert werden. Daher ist auf die Einhaltung der Werte nachfolgender Tabelle zu achten.

Mit der Firmware für diese Baugruppe ist die Anzeige folgender Informationen realisierbar:

| MC | Typ | Adresse | Anzeige | [ ] | Hinweis |
|---|---|---|---|---|---|
| M | Master - ATmega16(A) | - | | | Steuerrechner |
| S1 | Slave - ATmega8(A) | H22 - D34 | 7-Segment | hh | Stunden |
| S2 | Slave - ATmega8(A) | H24 - D36 | 7-Segment | mm | Minuten |
| S3 | Slave - ATmega8(A) | H26 - D38 | 7-Segment | ss | Sekunden |
| S4 | Slave - ATmega8(A) | H28 - D40 | 7-Segment | TT | Tag |
| S5 | Slave - ATmega8(A) | H2A - D42 | 7-Segment | MM | Monat |
| S6 | Slave - ATmega8(A) | H2C - D44 | 7-Segment | JJ__ | Jahr (Teil 1) |
| S7 | Slave - ATmega8(A) | H2E - D46 | 7-Segment | __JJ | Jahr (Teil 2) |
| S8 | Slave - ATmega8(A) | H30 - D48 | LED-Punkt* | WT | Wochentag |

*): Auswahlmöglichkeit zwischen Punkt- oder Bandanzeige
(ab Version V1.1.n)

H: hexadezimal
D: dezimal

Es ist stets darauf zu achten, dass der jeweiligen Anzeige die korrekte Adresse zugeordnet wird, da der Master die Daten nur an diese Andresse sendet.

Der Einsatz pinkompatibler MC ist möglich, erfordert allerdings, dass das Programm mit veränderten Einstellungen neu kompiliert werden muss.

Damit erfolgt z. B. die Anzeige (Zeit-Datum-Wochentag) in folgendem Format:

7-Segment (oben) 7-Segment (unten) LED-Punkte (senkrecht)

Eine mögliche Anordnung zeigt folgendes Bild:

Unten rechts befindet sich der Helligkeitssensor und die Statusanzeige (LED), mit der je nach Betriebsart u. a. der Hinweis auf Sommer-, Winter- oder UTC-Zeit angezeigt wird.

Da die Anzeigen in den unterschiedlichen Großen und Farben verfügbar sind, sollte man vor dem Aufbau der Treiber zunächst den optimalen Wert für die Segment-Widerstände ermitteln, da diese zusätzlich mit unterschiedlichen Flussspannungen einhergehen und eventuell ungewollte Helligkeitsunterschiede hervorrufen; diese können damit auch gewollt realisiert werden.
Dies wäre ebenfalls möglich, wenn für die Segmentanzeigen eine PWM-Steuerung eingesetzt wird, die in dieser Version nicht vorhanden ist.
Weiterhin können beim Einsatz von Anzeigen unterschiedlicher Hersteller störende Farbunterschiede auftreten.

Neben zusätzlichen LED für die Trennzeichen könne auch die integrierten Punktanzeigen (Segment 8 und ggf. 9) eingesetzt werden. Die Festlegung, was angesteuert werden soll, erfolgt z. B. bei der eingesetzten Leiterplatte per Lötpads.

## 5.3 Statusanzeige

Auf der Basisplatine besteht die Möglichkeit, eine RGB-LED anzuschließen. Durch sie können verschiedene Modis angezeigt werden:

**Blinken (3x kurz)**
Die Baugruppe wurde eingeschaltet oder neu programmiert und das Programm wird abgearbeitet

**Dauerlicht**
Die Dekodierung ist abgeschlossen, die Zeit wird angezeigt, wobei es sich um die mitteleuropäische Sommerzeit handelt.

**Blinken (Sekundentakt bei korrektem Empfang)**
Es zeigt den Signalausgang des DCF-Moduls an.
Ein unregelmäßiges Blinken deutet auf einen schlechten Empfang hin. Bei einem rhythmischen Blinken im Sekundentakt mit kurzer und langer Anzeigedauer wird das DCF-Signal empfangen.

**Blitzen (alle zwei Sekunden)**
Nach einer erfolgreichen Dekodierung des Zeitsignals ist aktuell keine Dekodierung möglich (Ausfall des DCF-Signals, Störung).

**Dauerlicht**
Die Dekodierung ist abgeschlossen, die Zeit wird angezeigt, wobei es sich um die mitteleuropäische Winterzeit handelt.

**Dauerlicht**
Die Dekodierung ist abgeschlossen, die Zeit wird angezeigt, wobei es sich um die UTC-Zeit handelt.

**Dauerlicht**
Es ist ein Programm-Modus eingestellt, der auf der LCD-Anzeige zusätzliche Informationen anzeigt:
PA 3: Anzeige ADC (Wert des Helligkeitssensors)
Der Wert ist mit dem Potentiometer bei unbeleuchteten Sensor auf 200 einzustellen.
PA 4: Anzeige PWM-Wertes (Helligkeitsteuerung der Anzeige)
Er zeigt, je nach Beleuchtung des Sensors, die Reaktion der Helligkeitsregelung (Werte 1 bis 99)

## 5.4 Bedienung

An die Masterbaugruppe können bis zu 9 Tasten angeschlossen werden, die BCD-kodiert eingelesen werden.

In diesem Projekt kommen aktuell drei Tasten zur Anwendung:
Taste 1 - Optionsauswahl
Taste 2 - Wert-Erhöhung
Taste 3 - Wert-Reduzierung

Ein Drücken der Taste 1 bewirkt das Fortschalten bei den Optionen, was durch folgende Anzeigen auf den beiden Digits (Slave H22) sichtbar wird.
Zusätzlich erfolgt eine Anzeige auf dem LCD-Display.
Der automatische Rücksprung zur Anzeige erfolgt nach ca. 5 Sekunden.

**1: Option PA - Programmauswahl**

Hier kann zwischen der Anzeige der MEZ (Sommer- oder Winterzeit) bzw. UTC umgeschaltet werden.
Weitere Modis zeigen den Wert des Helligkeitssensors sowie den PWM-Faktor der Anzeigehelligkeit.
Hinweis:
Die korrekte Darstellung des Datums beim Umschalten während des Tageswechsels ist in der Firmwareversion 0122 noch nicht geprüft.

Einstellmöglichkeiten ab der Programmversion V 1.0.30:
1: Normalzeitanzeige (MEZ)
2: Anzeige der UTC-Zeit (Winter: -1 h, Sommer: -2 h)
3: LCD-Anzeige mit eingelesenen Sensorwert
4: LCD-Anzeige der Helligkeitssteuerung (PWM)
5: -

In den Modis 3/4 leuchtet die Status-LED rot und für eine korrekte LCD-Anzeige des Sensorwertes (Einstellung 3) muss die Helligkeitssteuerung SH aktiviert sein.

**2: Option AC - Anzeigekostrast (LCD-Modul)**

Änderung des LCD-Kontrasts

**3: Option AH - Anzeigehelligkeit**
Änderung der Grundhelligkeit der gesamten Anzeige

**4: Option AL - Anzeigehelligkeit der Status-LED**
Helligkeitsanpassung der Status-LED

**5: Option SA - Slave-Anzahl**
Festlegung der Anzahl der angeschlossenen Slaves
Je nach Ausbau der Anzeige kann die Anzahl der Slaves angegeben werden, die beim Modulstart geprüft werden sollen.

**6: Option SH - Helligkeitssteuerung per Sensor**
automatische Anpassung der Anzeigehelligkeit
0: aus
1: ein

**7: Option EA - Einstellung der Anzeige**
Umschaltung der LCD-Jahresanzeige
0: DD:MM:JJ
1: DD:MM:JJJJ

**11-16: Option [ ] - Zeiteinstellung**
Ist ein DCF-Empfang für eine absehbar lange Zeit nicht möglich, kann manuell eine Zeiteinstellung vorgenommen werden:
H-: Einstellung der Stunden Zehner
-H: Einstellung der Stunden Einer
M-: Einstellung der Minuten Zehner
-M: Einstellung der Minuten Einer
S-: Einstellung der Sekunden Zehner
-S: Einstellung der Sekunden Einer

### 5.4.1 Speicherposition der Einstellungen

Die im Punkt „Bedienung“ gemachten Einstellungen sind auch nach einem Neustart der Anzeige unverändert, weil sie bei einer Änderung in das EEprom des MC geschrieben werden.
Daher ist es ebenfalls ohne eine Tastenbedienung möglich, die Optionen zu verändern, wenn man mit einem Programmiergerät Zugriff auf das EEprom hat.
Bei der Einstellung der Fuse- und Lockbits bei einem fabrikneuen MC ist daran zu denken, dass nicht ungewollt dieser Bereich bei

Softwareaktualisierungen überschrieben wird.

Die Adresse &H00 wird nicht verwendet, weil mir nicht bekannt ist, ob es noch beim Einsatz dieser Speicherzelle zu den im Internet beschriebenen Fehlfunktionen kommen kann.

Die Belegung des Speicherbereiches kann je nach Programmversion abweichen. Informationen dazu können daher direkt dem Quellcode entnommen werden, ob weitere Optionen verfügbar sind.

# 6 Zusatzbaugruppen

## 6.1 DCF-Simulator mit Option Stromschleife

Diese Baugruppe kann je nach gewählter Betriebsart (BA) die unterschiedlichsten Funktionen bereitstellen.
Sie ermöglicht die Überprüfung von Digitaluhren, die mit dem DCF77-Signal arbeiten, unabhängig von der aktuellen Zeit (Jahreswechsel, Umstellung auf Sommer- bzw. Winterzeit, Steuerung über Schaltzeiten).
Neben einem 3-Draht- kann ein 2-Drahtanschluss realisiert werden.

Der intern zum Einsatz kommende Festspannungsregler (3,3 V) kann bei Bedarf durch einen 5V-Typ ersetzt werden. Wenn es erforderlich ist, kann damit problemlos der Signalpegel an die sich anschließende Auswerteelektronik angepasst werden.

Weitere Optionen ergeben sich aus der Konfiguration vorhandener Lötpads.
Jeweils eine LED für die Anzeige des Zeitsignals sowie der BA und drei Taster komplettieren die Schaltung und können je nach Bedarf bestückt werden.
Optional kann statt einem Quarz mit der etwas größeren Bauart (8 MHz) ein kleiner Uhrenquarz mit 32,768 kHz eingesetzt werden. Die Quarzbeschaltung sowie die Software ist in diesem Fall anzupassen und - bedingt durch die geringere Taktfrequenz - im Funktionsumfang ggf. einzuschränken.

Die Schaltung ermöglicht es, dass Modul an einer Spannungs- oder Stromschleife zu betreiben, wenn die schaltungstechnischen Grenzwerte eingehalten werden.
Der Strombedarf von Basisplatine mit einem DCF-Modul liegt bei ca. 10 mA; ohne DCF-Modul ist mit ca. 6 mA zu rechnen. Die Werte wurden bei einer Betriebsspannung von 5 V ermittelt.
Bei den eingesetzten LED handelt es sich um hocheffiziente Typen. Bei anderen sind ggf. die Widerstandswerte anzupassen.

Durch einen MC (ATtiny84), dessen Software einschließlich dem Quelltext (BASCOM) kostenlos ist und über die vorhandene Programmierschnittstelle verändert werden kann, sind zurzeit folgende Einsatzfälle möglich:

**BA 0**:
Das Signal eines optional aufgesteckten DCF-Empfangsmoduls wird durchgeleitet. Intern wird es dekodiert und die Zeitinformation kann über ein ansteckbares serielles USB-Schnittstellenmodul an einen PC übertragen werden (getestet mit dem Terminalprogramm „CoolTerm“).
Durch einen Tastendruck kann die Phasenlage des weitergeleiteten Signals invertiert ausgegeben werden.

**BA 1:**
Hier arbeitet der Simulator ohne/mit einem externen DCF-Modul.
Beim Start wird intern eine Initialisierungsroutine durchlaufen, die eine festgelegte Zeitinformation in die intern laufende Uhr schreibt.
Aus ihr wird das Zeittelegramm generiert und ausgegeben.
Die Zeitinformation kann über ein serielles USB-Schnittstellenmodul an einen PC übertragen werden.
Auch hier kann über eine Taste die Phasenlage des weitergeleiteten Signals invertiert ausgegeben werden.

**BA 2:**
In dieser BA kann mit einem seriellen USB-Schnittstellenmodul eine Zeitvorgabe von einem PC übernommen werden.

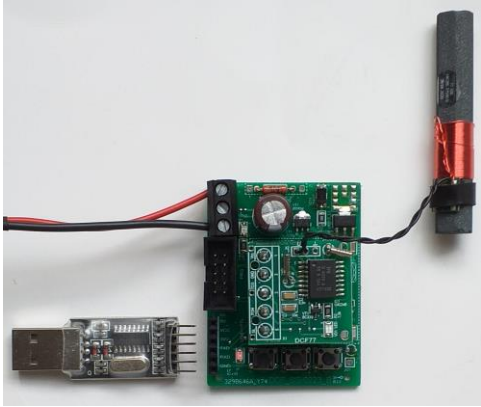
Einsatzmöglichkeiten
Manche Uhren „bleiben stehen“, wenn das DCF-Signal gestört ist.
Durch Zwischenschaltung des Moduls erfolgt unterbrechungsfrei die Übertragung der Zeitinformation durch die quarzgenau laufende interne Uhr.
Durch den MC ist es mit einer Softwareanpassung beispielsweise möglich, einen Minutenimpuls auszugeben, den bestimmte Uhren benötigen.

Bei der Leiterplatte handelt es sich um eine industriell hergestellte mit Durchkontaktierung sowie beidseitigen Beschriftungsaufdruck und Lötstopplack; sie erfordert den Einsatz von SMD-Bauteilen.

Der MC kann ggf. durch pinkompatible Typen ersetzt werden, was ggf. ein neues Kompilieren der Software voraussetzt.

Vorgestellt wird die Baugruppe unter folgendem Videolink:

https://youtu.be/8gweQAIoXUM



Hier die Ansicht der Simulatorbaugruppe mit einem aufgesteckten DCF-Empfangsmodul; links ein USB-Schnittstellenmodul zur PC-Kopplung.

# 7 Ausführungen

## 7.1 Anzeige über Grafikdisplay

https://youtu.be/PjtPocO3sFs

Unterhalb der Displays befindet sich eine Platine, auf der sich der jeweilige Slave befindet.
Er beinhaltet die Software, die die Form, Farbe und Größe der anzuzeigenden Daten bestimmt.

Für diese Art der Anzeige ist eine verkleinerte Basisplatine verfügbar.

## 7.2 Anzeige mit 14/16-Segment-Anzeigen

https://youtu.be/QKVw55K2zO8

Noch kompakter wird der Aufbau mit dieser Version.

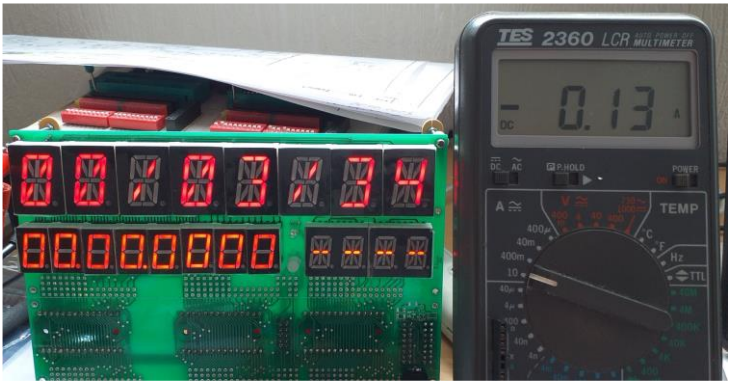
Mit dem Einsatz von SMD-Bauelementen auf der Basisplatine und drei speziellen Anzeigeschaltkreisen (HT16K33) wird der Platzbedarf der Schaltung auf ein Minimum reduziert.
Die Betriebsspannung für die Anzeige ist allerdings durch den IC auf 5 V festgelegt.

Der Platzbedarf der Front kann verringert werden, wenn die Platine an der dafür vorgesehene Stelle geteilt wird. Die erforderlichen Verbindungen sind danach mit Steckverbindern nachzurüsten.

Neben der etwas gefälligeren Anzeige, die die 14/16-Segmente der Anzeigeelemente ermöglichen, ist eine Soundausgabe implementiert. Über sie kann eine Zeitansage und/ oder ein Sound (Gong) zu bestimmten Zeiten ausgeben werden.

Die Basisplatine ermöglicht die Auswertung von zwei Temperatursensoren, wobei sich einer auf der Basisplatine befinden kann.

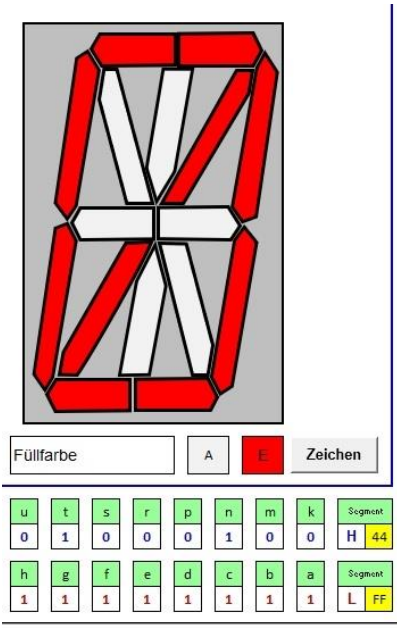
Verfügbare Adapterplatinen ermöglichen den Einsatz von 7-Segment-Anszeigen für die Zeitanzeige.



# 8 Tools

## 8.1 Kodierung für 7-, 14- und 16-Segmentanzeigen

Für die Darstellung mit Segmentanzeigen ist im Datenbereich des MC eine Segment-Kodierung erforderlich.
Für die Erstellung dieser Kodierung kann das jeweilige Registerblatt einer Excel-Applikation zum Einsatz kommen.
Neben dem bereits eingetragenen kann der Kode für jede Art von Sonderzeichen ermittelt werden.




| ASCI | Z | Code | | |
|---|---|---|---|---|
| | | [Hex] | H [Bin] | L |
| 033 | ! | 0000 | 00000000 | 00000000 |
| 034 | " | 0204 | 00000010 | 00000100 |
| 035 | # | AA3C | 10101010 | 00111100 |
| 036 | $ | AABB | 10101010 | 10111011 |
| 037 | % | EE99 | 11101110 | 10011001 |
| 038 | & | 9371 | 10010011 | 01110001 |
| 039 | ' | 0200 | 00000010 | 00000000 |
| 040 | ( | 1400 | 00010100 | 00000000 |
| 041 | ) | 4100 | 01000001 | 00000000 |
| 042 | * | FF00 | 11111111 | 00000000 |
| 043 | + | AA00 | 10101010 | 00000000 |
| 044 | , | 4000 | 01000000 | 00000000 |
| 045 | - | 8800 | 10001000 | 00000000 |
| 046 | . | 1000 | 00010000 | 00000000 |
| 047 | / | 4400 | 01000100 | 00000000 |
| 048 | 0 | 44FF | 01000100 | 11111111 |
| 049 | 1 | 040C | 00000100 | 00001100 |
| 050 | 2 | 8877 | 10001000 | 01110111 |
| 051 | 3 | 083F | 00001000 | 00111111 |
| 052 | 4 | 888C | 10001000 | 10001100 |
| 053 | 5 | 90B3 | 10010000 | 10110011 |
| 054 | 6 | 88FB | 10001000 | 11111011 |

Dieses Bild zeigt die Bedienoberfläche für eine 16-Segmentanzeige.

Die Arbeitsweise mit diesem Makro wird unter folgendem Videolink vorgestellt:

https://youtu.be/tzqcZ2IliHU

Dieses Tool ist für den Einsatz der Anzeigen vom Typ PSA08-11 ausgelegt.
Es ist in einer Testversion verfügbar, bei der die Kodier-Tabelle für die Anzeigen vom Typ VQB201 modifiziert wird. Dazu kommen spezielle Funktionen in Excel zum Einsatz, da mit sehr großen Binär- und HEX-Werten gerechnet wird.
Damit wird es möglich, durch die Änderung des Makros für die „Umkodierung“ den Segment-Code an weitere Anzeige-Typen anzupassen.