Microcontroller für Einzelanwendungen

Hans Peter Stoehrel, Universität Stuttgart

DECUS Symposium 2003 in Bonn

Inhalt

Microcontroller finden sich heute nicht nur in der Waschmaschine und im Staubsauger, auch Babys Flaschenwärmer enthält einen solchen Vielfüssler. Das sind aber alles Großserienprodukte, bei denen sich der hohe Aufwand für die Programmierung, die hier fast durchwegs aus ökonomischen Gründen in Maschinensprache erfolgt, lohnt. Hat man aber nur eine einzelne Aufgabe vor sich, die etwas umfangreichere logische Verknüpfungen erfordert, Messungen vornehmen soll oder Steuerungen erledigen soll, so stellt sich die Frage der Effektivität. Hier können Microcontroller eingesetzt werden, die in BASIC (ja, BASIC!!) programmiert werden und daher schnell einsatzbereit sind. Trotzdem lassen sich mit ihnen hochkomplexe Anwendungen realisieren.
Der Vortrag soll eine kleine Einführung in zwei marktgängige Systeme (BASIC-Tiger von Wilke und C-Control von Conrad) geben. Es wird Gelegenheit gegeben, die zugehörigen Entwicklungssysteme selbst in die Hand zu nehmen und auszuprobieren, außerdem werden Beispielsysteme gezeigt.

Einleitung

Beginnt man mit den Überlegungen zu einem Microcontroller-System, so stehen oft die Überlegungen, wie das Programm im Controller auszusehen hat, im Vordergrund. Aber das ist nach unserer Erfahrung nicht der richtige Anfang. Natürlich gehört einige Erfahrung dazu, den richtigen Controller anhand seiner Eigenschaften (und Eigenheiten!) und das Programm-Design dazu auszusuchen, aber es stellt sich schnell heraus, dass doch die Elektronik an erster Stelle überlegt werden muss, dann die Mechanik, in die das Ganze eingepackt werden muss, und die Programmierung kommt dann fast von selbst.
Als außerordentlich hilfreich erweist sich immer ein gut konstruiertes Hardware-Entwicklungssystem, dass man auf jeden Fall zunächst zum Testen der gewünschten Funktionen einsetzen sollte. Beide Entwicklungssysteme der hier beschriebenen Microcontroller sind gut brauchbar, im Fall der C-Control haben wir sogar ein Entwicklungssystem als Zielsystem verwendet und komplett „vergraben“.

Hardware-Entwicklungssysteme
BASIC-Tiger C-Control

Software-Entwicklungssysteme
Beide Microcontroller werden mit guten Programmierumgebungen geliefert. Sie enthalten jeweils einen Programmeditor, einen Compiler und einen Downloader, der das fertige Programm in den Microcontroller „hinunterlädt“. Das Tiger-System enthält außerdem noch einen Debugger. Die Downloader können auch separat verwendet werden, damit kann man auch übersetzte Programme an Kunden versenden, die sie dann damit selbst in den Microcontroller laden können.

Übersichtstabelle Microcontroller

	C-Control	BASIC-Tiger
Allgemeines:		
Hersteller	Conrad Electronic	Wilke Technology
Web-Seite	www.conrad.de	www.wilke-technology.com
Entwicklungssystem Hardware	Ja	Ja
Entwicklungssystem Programm.	Ja	Ja
Erf. PC für Programmentwicklung	DOS-PC, Windows	WIN98 aufwärts
Downloader separat	Ja	Ja
Kommunikation mit PC	RS232 (COM) einstellbar	RS232 (COM) einstellbar
Downloadgeschwindigkeit	Mittel	Niedrig
Debugger online	Nein	Ja
Gerätetreiber (LCD, Tastatur etc)	Nein	Ja
Passwortschutz gegen Auslesen	Nein	Ja
Dokumentation	Ausreichend	Sehr gut
Beispielprogramme	Brauchbar	Sehr gut
Hardware:		
Prozessor	MC68H05B6	Unbekannt(wird nicht mitgeteilt)
Taktfrequenz	4 MHz	ca. 20 MHz
Arbeitsgeschwindigkeit System	Niedrig	Sehr hoch
Speicher RAM	24 Byte !!!	128 kB...2 MB je nach Typ
Speicher Flash	8 kB	128 kB...4 MB je nach Typ
I/O-digital (PINs)	16	Max. 33 je nach Typ
Bitweise als Ein/Ausgänge	Ja	Ja
Belastbarkeit dig. Ausgänge	10 mA	1.6 mA
I/O analog (A/D-Einänge)	8x 0 – 5 V, 8 Bit	4x 0 – 5V, 8 oder 10Bit
PWM	2 (max. 1.9 kHz)	2x, 6-, 7-, 8 Bit, 1.2-80kHz
Hardwareerweiterungen	Nein	Ja
Frequenzmessung	Ja, max. 32kHz	Ja, max. 1.2 MHz
Pulslängenmessung	Bedingt	Ja, min. messbare Pulslänge 0.4 µs
RS232 für externe Geräte	Ja, externer Wandler	Ja 2x, intern. oder externer Wandler
DCF-Dekodierung	DCF-Dekoder integeriert	DCF-Dekoder per Software
Sleep	Ja	Ja
Interne Echtzeituhr	Ja	Ja (Option je nach Typ)
Wake-Up intern/extern	Ja	Ja
Interrupt	Ja	Nein
Stromversorgung	5 V ca. 6 mA	5 V ca. 50 mA
Pinning	39 Pin spezial quadratisch	DIL spezial 2x23 je nach Typ
Mech. Abmessungen	43 x 40 x 8 mm +Pins	63 x 41 x 12 mm + Pins
Programmierung:		
Programmiersprache	Basic	Basic
Assembler	Ja	Nein
Multitasking	Nein	Ja
Subroutinen	Ja	Ja
Gleitkommaarithmetik	Nein	Ja
Funktionen:		
Mathematisch	nur Integer-Arithm.	volle Gleitkomma-Arithm., trig. Funkt.
String	Ja	Ja
Bit	Ja	Ja
Variable:		
Byte	Ja	Ja
Integer 16 Bit	Ja	Ja
Integer long (32 Bit)	Nein	Ja
Real	Nein	Ja (8 Byte, ca. $\pm 10^{308}$)
String	Ja	Ja , max. Länge 65kB
Arrays	Nein	Ja
Flashdaten schreiben/lesen	Ja	Ja

Microcontroller: links BASIC-Tiger, rechts C-Control

Allgemeine Design-Hinweise

Im folgenden werden stichwortartig einige Hinweise für die Entwicklung gegeben, die sich bei uns als hilfreiche Richtlinien bei unseren Projekten erwiesen haben.

Hardware

- Anschlüsse für Ein- und Ausgänge an Maschinen oder andere größere Anlagen sollten nur über Optokoppler (bei digitalen Ein/Ausgängen) bzw. über Pufferverstärker (OpAmps) oder, besser, über galvanisch isolierte Trennverstärker (bei analogen Ein/Ausgängen) erfolgen. Der Aufwand ist zwar hoch, aber die Systemsicherheit steigt. Wenn irgend etwas Böses passiert, ist es wesentlich einfacher die Trennglieder zu tauschen (so sie gesockelt sind!) als den Prozessor.
- Die Stromversorgung muss sorgfältig entkoppelt werden, man sollte möglichst ein eigenes Netzteil für den Microcontroller und dessen unmittelbare Peripherie vorsehen.
- Zum einfacheren Testen der Eingänge sollten Pins zum Messen der Eingangsspannungen (analog und digital) vorgesehen werden. Testpins für Masse und Betriebspannung nicht vergessen!
- Eine Kontrollanzeige für das laufende Programm sollte vorhanden sein (LED)

Software

- Sehr wichtig ist eine sorgfältige Programminitialisierung wegen des Selbststarts des Programms beim Einschalten.
- Um einem „Aufhängen“ des laufenden Programms zu begegnen, müssen Maßnahmen wie etwa ein Watchdog implementiert werden. Dies ist besonders wichtig, wenn der Microcontroller irgendwo vergraben wird und normalerweise nicht bedient wird. Auf jeden Fall muss auf der Platine ein Anschluss (Jumper) für einen Reset vorgesehen werden.

Bemerkungen zum BASIC-Tiger

Der BASIC-Tiger weist eine bemerkenswerte Eigenschaft auf: er ist voll multitaskingfähig. Man kann bis zu 50 Tasks parallel ablaufen lassen und jeder Task eine eigene Priorität zuweisen. Tasks können sich gegenseitig aufrufen und auch anhalten. In Schleifen kann man überschüssige Wartezeit freigeben. Damit hat mein System, das auch für anspruchsvollere Steuer- und Regelsysteme einsatzfähig ist.

Ausgeführte Projekte mit Microcontrollern (Auswahl)

Koffer-Heizgerät für Dialysebeutel

Die Dialyse für Nierenkranke kann nicht nur im Krankenhaus, sondern auch zu Hause durchgeführt werden. Dafür werden Beutel mit einer speziellen Flüssigkeit verwendet, die der Patient über einen Anschluss regelrecht an seinen Bauch anschließt. Das Bauchfell wirkt dann als Austauschfilter.
Die Beutel fassen ca. 1 ltr. Flüssigkeit, sie müssen vor Verwendung unbedingt auf Körpertemperatur gebracht werden, da der Patient sonst einen Temperaturschock erleidet. Normalerweise werden die Beutel auf einer netzbetriebenen Heizplatte vorgewärmt, deren Gebrauch ist aber unterwegs auf Reisen nur sehr schwierig zu bewerkstelligen. Daher wurde von uns ein Alu-Werkzeugkoffer ausgeräumt, wärmetechnisch isoliert und mit einer selbstgebauten Heizplatte versehen. Ein Netzgerät

für 220V sowie ein Autoanschlussstecker mit entsprechendem Netzgerät fanden ebenfalls Platz im Koffer.
Nun mussten nur noch die Temperaturregelung und eine Zeitschaltuhr eingebaut werden und das Gerät war fertig. Als Microcontroller wurde eine C-Control als ausreichend angesehen.
Das Pflichtenheft sah wie folgt aus:

- Heizung mit Leistungswiderständen: 5x6.8 Ω 25 W => 100 W Heizleistung (bei 12V ca.8A)
- Versorgung durch Autobatterie 12 V oder 220V-Netz mit Halogentrafo 12 V /105W, umschaltbar
- Temperaturregelung auf 37°C und Zeiteinstellung mit C-Control
- 2 Temperaturfühler KTY10 zur sicheren Temperaturmessung und Schutz gegen lokale Übertemperaturen
- Temperatur-Sicherung über getrennten PTC (reine Hardwaretechnik, nicht über die C-Control)
- Anzeigen:
  Rot (blinkend): Übertemperatur (Hardware)
  Rot (Dauer): Betriebsspannung zu niedrig
  Grün: Beutel hat eingestellte Temperatur erreicht
  Gelb: Heizt
- Zeitvorwahl: mit Stufenschalter 10..12 Stufen, Abstand 1h
- Starttaste zum Heizbeginn

Das C-Control Programm umfasst ca. 250 Zeilen, davon 130 Zeilen Code. Das System benutzt zwei A/D-Kanäle, 11 dig. I/O (5 als Ausgänge, 6 als Eingänge benutzt), es benötigt ganze 4 (i.W. vier!) Byte als Variablen. Der print-Befehl sendet Daten auf die serielle Schnittstelle als Testausgabe.

Programmauschnitte:

```
....
' --- Definitionen Ports und Variablen -------------------
  define ntcfalte ad[4]       ' Messkanal "Falte"
  define ntcblech ad[3]       ' Messkanal "Blech"
  define relais port[1]       ' Ansteuerung Heizrelais
  define rot port[2]          ' Uebertemperatur bzw. Bruch der Fuehlerleitung
  define gruen port[3]        ' System laueft
  define gelb port[4]         ' heizt
.....
  define sw1 port[9]          ' Sollwerte der Blech- bzw. Faltentemperatur
  define sw2 port[10]         ' s. Tabelle am Programmende
  define tkomp port[15]       ' Prueft ob Komparator schaltet
  define testst port[16]      ' Testtaster
  define sollblech byte       ' 4 Variablen
  define sollfalte byte
  define swsollblech byte
  define swsollfalte byte
.....
#loop1                        ' Regelschleife

      if testst = 0 then goto testkomp

      if ntcblech > 250  then goto bruch   ' Temp-Fuehlerbruch?
      if ntcfalte > 250  then goto bruch
      if ntcblech > sollblech then relais = on else relais = off
      print  "loop1",sollblech, ntcblech, sollfalte, ntcfalte, abs(relais)
'      print " ",sw1,sw2,sw3,sw4,sw5,sw6
      gelb = on                       ' gelbe Anzeige-LED anschalten
      if relais = off then goto loop2  ' naechste Regelschleife
      goto loop1
......
```

Die für die Regelung relevanten Temperaturen wurden in einer Tabelle abgelegt, die direkt in Bytes (0...255) ausgedrückt sind. Damit erübrigt sich jede Umrechnung „Messwert (in Bytes)“ in „Temperatur (in °C).

Der Koffer wurde gleich auf eine längere Reise nach Italien mitgenommen und funktionierte zur vollsten Zufriedenheit der Patientin – eine Reise, die ohne das Gerät sehr beschwerlich gewesen wäre.

Messgerät für Membranspannungen

Im Bauwesen werden häufig Membranen als Bedachungen gewählt, weil sie architektonisch reizvoll aussehen und im städtischen Umfeld eine Abwechslung bieten, außerdem erlauben sie große Spannweiten bei geringem Gewicht.
Bei der Aufstellung eines Membrandaches muss die vorher durch den Statiker berechnete Vorspannung genau eingehalten werden, da davon die Beständigkeit des Daches gegenüber Wind- und Schneelasten abhängt.
Zur Messung der Membranspannungen gibt es verschiedene Verfahren. Ein sehr einfaches, aber in der Konstruktion des Messgerätes sehr anspruchsvolles Verfahren besteht darin, mit einem kleinen Hammer einen Schlag auf die Membran auszuführen. Mit dem Schlag läuft eine Körperschallwelle durch Membran, und deren Laufgeschwindigkeit ist direkt mit der Membranspannungen nach einer e-Funktion korreliert. Die Laufzeit der Welle wird mit zwei kleinen Beschleunigungsaufnehmern gemessen, von denen einer direkt neben dem Hammer und der andere in einer gewissen Entfernung an der Membran angebracht werden. Um das Gerät handlich zu halten wird der Abstand der beiden Aufnehmer auf ca. 200mm festgelegt. Die Laufzeiten betragen dann bei den verwendeten Membranen je nach Vorspannung zwischen 6 msec (geringe Vorspannung) und 1.5 msec (hohe Vorspannung.

Das Pflichtenheft für das Messgerät sieht wie folgt aus:

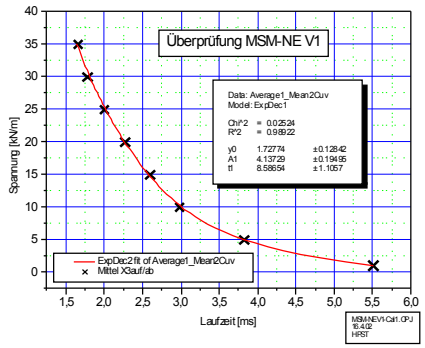
- es muss einen Schlag mit definierter Stärke und Dauer (ca. 300ms) ausgeführt werden
- es muss eine Zeitmessung gestartet werden, sobald der Beschleunigungsaufnehmer neben dem Schlaghammer anspricht
- die Zeitmessung muss gestoppt werden, sobald der zweite Beschleunigungsaufnehmer anspricht
- die Auflösung der Zeitmessung sollte für ausreichende Messgenauigkeit bei ca. 5µsec liegen.
- Die Messwerte müssen umgerechnet und in geeigneter Form ausgegeben werden.
-

Alle Punkte werden durch ein kleines BASIC-Tiger System voll erfüllt. Die Schlagauslösung stellt kein Problem dar. Die Messung der Impulswellen-Laufzeit ist auch kein Problem, die Auflösung kann bis zu 0.4µsec betragen bei einem Messbereich von 0.4µsec bis 26 msec. Der Programmteil zur Impulsmessung umfasst dank des Device-Treibers, der zum System gehört, ganze 4 Zeilen benötigt die Pulslängemessung: 1 Zeile zum Laden des Device-Treibers, 2 Zeilen zur Initialisierung (Messbereich) und 1 Zeile zur eigentlichen Messung.

Programmausschnitte:

```
TASK MAIN
  WORD XLEN, YLEN                  ' Variablen deklarieren
  REAL pulselength
  REAL xlenr, ylenr
  BYTE dacout, bitbit, nzl
  INSTALL DEVICE #1, "LCD1.TDD"        ' LCD-Treiber installieren
  INSTALL DEVICE #8, "PLS1IN.TDD", 1   ' Pulslaengentreiber / 1 = Bereich
...
' Pulslängenmessung aktivieren
  PUT #8,#0, #UFCO_IPL_RNG, 1  'Bereich 1(hoechste Aufloesung)1 digit=0.4µs
  PUT #8,#0, #UFCO_IBU_ERASE, 1    ' Lösche Eingangspuffer
...
' Pulslänge messen
  GET #8, 2, XLEN             ' Pulslaenge auslesen, liest 0 wenn kein Puls
  xlenr = XLEN * 0.4 / 1000          ' Umrechnen in ms
  print_using #1, "<1>X ="; xlenr    ' Testausgabe
```

Das folgende Bild zeigt die hervorragende Genauigkeit des Messgerätes. Die Kreuze stellen die Punkte dar, die das Messgerät bei einer eingestellten Vorspannung einer Probemembran lieferte. Die Güte zeigt sich dadurch, dass alle Kreuze auf der vorher berechneten e-Funktion liegen.



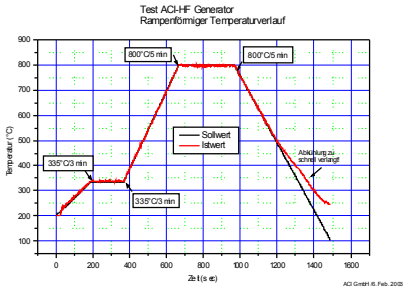
Kalibrierkurve für Membranspannungsmessgerät

Controller für Induktionsheizung

Induktionsheizungen werden z.B. zum gezielten Härten von Werkzeugen oder allgemein in der Materialprüfung benutzt. Dabei müssen die hohen Temperaturen (bis 1400°C und mehr), die mit diesen Geräten erreicht werden können, in genau vorgegebenen Zeiten gefahren werden, genau so muss anschließend der Abkühlvorgang gesteuert werden. Dazu werden als Vorgabe je nach Material und gewünschten Eigenschaften Rampenkurven benutzt, die der Controller nachfahren muss. Die Temperaturmessung geschieht berührungslos mit Infrarot-Pyrometern.
Für ein solche Induktionsheizung wurde ein Controller entworfen und aufgebaut, der folgendem Pflichtenheft genügt:

- Eingabe einer beliebigen Rampenkurve mit bis zu 50 Eckpunkten (jeder Eckpunkt besteht aus einem Wertepaar bestehend aus einer Zeit und einer Temperatur) mit Korrekturmöglichkeit auf einer Tastatur am Controller
- Dauer einer Rampenkurve bis zu mehreren Tagen, Zeiteingabe in Minuten
- Abspeicherung von bis zu 50 solcher Kurven im Controller, Wiederaufruf einer Kurve
- Nachfahren der Rampe mit einer Genauigkeit von ca. ±5°C (Pyrometergenauigkeit)
- Fortlaufende Ausgabe der Zeit, der Soll- und der Ist-Temperatur auf dem LCD-Display des Controllers und auf der seriellen Schnittstelle (Anschluss eines PCs zur Registrierung)

Als Controller wurde ein BASIC-Tiger gewählt, ausgestattet mit einer 4x4-Tastatur zur Eingabe und einem 4-Zeilen LCD-Display zur Ausgabe der Werte. Die gewünschte Regelgenauigkeit wurde problemlos erreicht, obwohl im Programm nur ein simpler 2-Punkt-Regler eingesetzt wurde, der die Induktionsheizung zwischen einem oberen und einem unteren Leistungswert hin- und herschaltet.



Induktionsheizung bei 1499°C
(Anzeige am Pyrometer links oben)

Beispiel einer Rampenkurve mit Regelung
(Abkühlung zu steil verlangt!)