Elektronikpraktikum

Microcontroller

Namen Lutz Bakemeier
Stephan Richter

Datum 09. Juli 2007

Versuchsgruppe M1

Grundlagen

Unter Mikrocontrollern versteht man Ein-Chip-Rechnersysteme, die insbesondere für Regelungs- und Steuerungsaufgaben eingesetzt werden. Im Gegensatz zu Microprozessoren, enthalten sie neben dem Prozessor noch weitere Elemente wie z.B. Arbeitsspeicher, Ein- und Ausgabe- Schnittstellen oder einen Taktgenerator. Ihre universelle Einsetzbarkeit stützt sich auf ihre Programmierbarkeit. Mikrocontroller sind RISC-Prozessoren (reduced instruction set computing). Dies hat den Vorteil eines geringeren Decodierungsaufwandes und führt zumeist zu schnelleren Programmen.

Zur Speicherung des Programmes und Daten lassen sich verschiedene Speicherarten verwenden. Prinzipiell kann man zwischen wiederberschreibbaren und nicht wiederbeschreibbaren Speichern unterscheiden. Zu ersteren gehören sogenannte EPROM. Sie können Informationen bis zu 40 Jahre lang speichern. Gelöscht werden können sie jedoch nur durch UV-Bestrahlung. Bei Massenproduktion verzichtet man zumeist auf diese Option, was die Kosten erheblich senkt. Die PICs können dann jedoch nicht für andere Aufgaben umprogrammiert werden. Benötigt man umprogrammierbare PICs, so müssen sogenannte EEPROM verwendet werden. Diese lassen sich unkompliziert löschen und bis zu 1 Million mal wiederbeschreiben. Die Informationen werden bis zu 10 Jahren sicher gespeichert, bei sogenannten Flash-EEPROM bis zu 40 Jahren.

Aufgabenstellung

1 Problemstellung

Es soll ein Servomotor, der eine impulslängen-modulierte Steuerspannung benötigt, mit einer analogen Eingangsspannung zwischen 0 und 5V angesteuert werden.

Die Impulsdauer sollte dabei von $0,65ms$ bis $2,75ms$ variiert werden, was einer Auslenkung von 0 bis 180° entspricht. Die Impulsfrequenz sollte etwa $50Hz$ entsprechen.

2 Aufgabenstellung

Die Aufgabe war es, dass oben beschriebene Problem mit Hilfe eines ADC 0804 und eines PIC 16F84 zu lösen und die Funktionalität unseres Systems zu testen.

Realisierung

3 Aufbau

Die zur Realisierung benötigte äußere Beschaltung erfolgte nach der Abbildung 1. Als Quelle zum

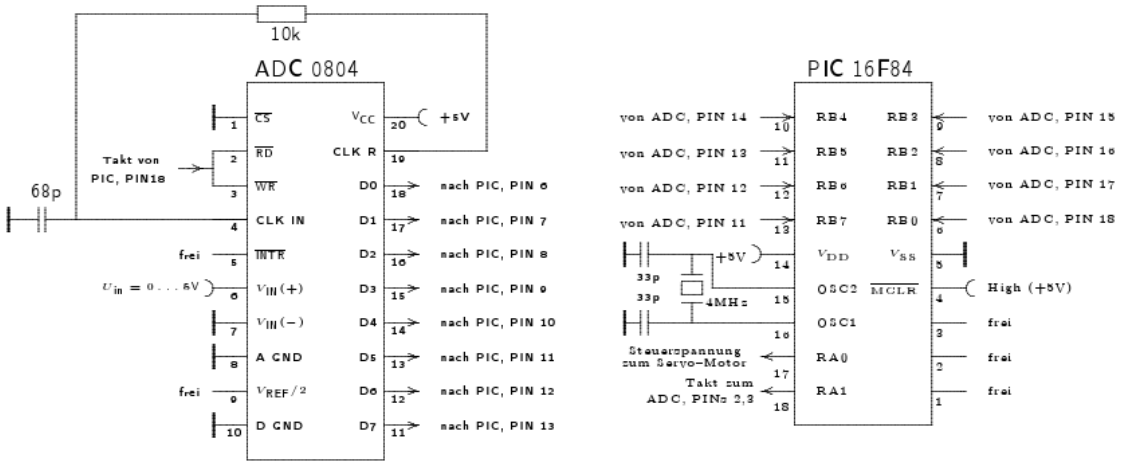

Abbildung 1: Beschaltung des ADC und PIC

Testen des Aufbaus wurde der Funktionsgenerator bzw. das Stromversorgungsgerät verwendet. Letzteres lieferte auch die Betriebsspannung von PIC, ADC und Servo-Motor. Mit dem Oszilloskop prüften wir ob die programmierten Schleifen die richtigen Impulsdauern lieferten.

4 Programmierung

Im folgenden wird der Quelltext hinsichtlich seiner Funktionalität beschrieben.

```
; Servomotor ansteuern
; ================================================================
;--------------------------------------------------------------------------------
; Initialisierung des PIC
;--------------------------------------------------------------------------------

  #include "p16f84.inc"                    ; Bezeichner- und Konstantendefinition
  LIST     p=pic16F84                      ; Auswahl des Mikrocontrollers
  __CONFIG _XT_OSC & _PWRTE_OFF & _WDT_OFF  ; Takterzeugung durch externen Quartz
```

Im ersten Programmteil wird die Include- Datei für den verwendeten Mikrocontroller eingebunden. Sie enthält hauptsächlich equ- Befehle, welche häufig verwendete Adressen einen Namen zuweisen, was die Programmierung und das Lesen des Quelltextes erheblich vereinfacht. Die letzte Zeile teilte dem PIC mit, von wo der Takt geliefert wird.

```
;-------------------------------------------------------------------------------
; Initialisierung der Ports
  BSF STATUS, RP0               ; Auswahl von bank 1
  MOVLW       B'00000000'       ;
  MOVWF       TRISA             ; Festlegung: RA4..RA0 sind Ausgänge
  MOVLW       B'11111111'       ;
  MOVWF       TRISB             ; Festlegung: RB7..RB0 sind Eingänge
  BCF STATUS, RP0               ; Auswahl von bank 0
```

In diesem Programmabschnitt werden die Ein- und Ausgabeports deklariert. Um dies zu erreichen, müssen die Adressen TRISA und TRISB angesprochen werden. Diese befinden sich in der Bank 1. Um dahin zu wechseln wird das STATUS Bit mit dem Befehl BSF (bit set file) gesetzt. Da wir die Pins RA4 bis RA0 als Ausgänge verwenden wollen, wird TRISA entsprechend Low gesetzt. Dies muss über das Operandenregister geschehen. Dazu wird die Zeichenkonstante B'00000000' mit dem Befehl MOVLW in das Operandenregister W geladen. Mit dem Befehl MOVWF, wird der Inhalt von W in das File TRISA überschrieben. Das selbe geschieht mit TRISB, nur das hier High gesetzt wird. Der Wert im TRIS Speicher bestimmt ob der entsprechende Pin hoch- oder niederohmig ist. Zum Schluss wechseln wir wieder auf Bank 0 mit dem Befehl BCF (bit clear file).

```
;-------------------------------------------------------------------------------
; Hauptprogramm
;-------------------------------------------------------------------------------
mainloop

  BSF PORTA,1 ;ADC übergibt werte
  CALL ADCDelay ;Warten bis Werte an Ausgang von ADC liegen
  MOVF PORTB,0 ;Werte vom ADC in das Workingfile
  MOVWF 0x0E ;Werte in 0x0E
  BCF PORTA,1 ;Datenübernahme beendet
  INCF 0x0E,1 ;Erhöhung um 1 zur Verhinderung des Überlaufs bei Null

  BSF PORTA,0 ;Setzen von Ausgang auf High
  CALL SDelay ;Duchlaufen der StandardSchleife
  CALL VarDelay ;vom Eingang abhängige Schleife
  BCF PORTA,0 ;Zurücksetzen des Ausgangs
  CALL BDelay ;20 ms Schleife
GOTO mainloop
```

Zu Beginn dieses Abschnitts befindet sich die Marke mainloop, die für das Springen an diese Stelle nach dem Schleifendurchlauf benötigt wird. Innerhalb der Schleife werden folgende Befehle sequentiell abgearbeitet:

- Mit dem ersten Befehl wird der ADC angesprochen, welcher daraufhin das Abtasten einstellt und die derzeitige Digitalwort an seine Ausgänge legt.
- Da dies nicht sofort geschieht, muss der PIC warten. Dies wird durch den Aufruf (call) einer Verzögerungsschleife von $18\mu s$ erreicht.
- Nach Durchlauf dieser Schleife beginnt die Verarbeitung des Digitalwortes. Mit MOVF wird die Belegung des PORTB in das Operandenregister geschrieben.
- Dessen Inhalt wird anschließend zur späteren Verwendung in ein Register geschrieben.
- Mit dem Rücksetzen des Bit 1 des PORTA wird der ADC zurück gesetzt. Die Datenübernahme ist beendet.
- Durch den Befehl INCF wird der Wert des übergebenen Digitalwortes um 1 erhöht, was später das Überlaufen in der Schleife VarDelayLoop1, für den Wert 0 verhindert.

Nun wird aus dem übernommenen Digitalwort die Steuerspannung generiert. Dazu wird zunächst der den Servomotor steuernde Ausgang auf High gesetzt.

- Anschließend werden nacheinander zwei Schleifen aufgerufen. Die erste erzeugt die obligatorischen $0,65ms$, die zweite eine vom Wert des Digitalwortes abhängige Zeitdauer zwischen 0 und $2,1ms$.
- Nach Durchlaufen dieser wird der Ausgang wieder rückgesetzt, was dem Ende des Pulses entspricht.
- Nun wird eine weitere Schleife aufgerufen, die eine Verzögerung von $20ms$ hervorruft, was die geforderte Frequenz von $50Hz$ realisiert.
- Mit dem Befehl GOTO springt das Programm zurück zur Marke, so dass die Schleife erneut durchlaufen wird.

```
;-------------------------------------------------------------------------------
; Unterprogramme

VarDelay                    ; Variable Scleife
VarDelayLoop1
      NOP
      NOP
      NOP
```

```
      NOP
      NOP
      DECFSZ 0x0E, 1
      GOTO VarDelayLoop1
RETURN
```

- Die in der Hauptschleife aufgerufenen Unterprogramme müssen mit der entsprechenden Marke beginnen, in diesem Fall VarDelay.
- Die nächste Zeile ist die Marke für die folgende Schleife.
- Der NOP-Befehl dient zur Verzögerung, was in diesem Fall einer Skalierung zwischen Digitalwort und Durchlaufzeit entspricht.
- Die Größe des Digitalwortes bestimmt die Abbruchbedingung. Dies wird realisiert durch den Befehl DECFSZ. Dieser verringert pro Aufruf den Wert des nachgestellten Registers um 1. Wird dieser Wert Null, so wird der nachfolgende Befehl durch ein NOP ersetzt. In diesem Fall bedeutet das den Abbruch der Schleife.
- Der Befehl RETURN weist das Programm an mit dem nächsten Befehl im Hauptprogramm fortzufahren.

```
ADCDelay                      ; Scleife für 18 us
      MOVLW 0x06                  ; 1 cycle
      MOVWF 0x0F                  ; 1 cycle
ADCDelayLoop1
      DECFSZ 0x0F, 1arDelayLoop1
      GOTO SDelayLoop1
RETURN

SDelay                      ; Scleife für 0,65 ms
      MOVLW 0xD4                  ; 1 cycle
      MOVWF 0x0C                  ; 1 cycle
SDelayLoop1
      DECFSZ 0x0C, 1
      GOTO SDelayLoop1
RETURN

BDelay                      ; Scleife für 20 ms
      MOVLW 0x23                  ; 1 cycle
      MOVWF 0x0D                 ; 1 cycle
BDelayLoop1
      CALL SDelay
```

```
      DECFSZ 0x0D, 1
      GOTO BDelayLoop1
RETURN

END
```

Diese Schleifen entsprechen der oben beschriebenen, mit dem Unterschied, dass der zu dekrementierende Wert schon feststeht.

Auswertung

Nach Behebung verschiedener Fehler konnte die Funktionalität des Systems erfolgreich überprüft werden. Es war jedoch zu beobachten, dass bei Frequenzen höher $1Hz$ der Servomotor zu träge war um dem Wechsel der Steuerspannung zu folgen. Dies zeigte sich darin, dass die anvisierte maximale Auslenkung nicht mehr erreicht wurde.

- Pompe: *Praktikumsanleitung.* Greifswald [Stand: 29.06.2006]
- Microchip: *Datenblatt des PIC16F84.*