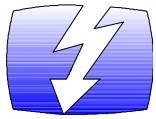

Prof. Dr.-Ing. Kühne **April 2005**
Bearb.:Dipl.-Ing. Sbieschni

Versuch 1: **Ein-Ausgabe**

Zielstellung:

- Kennenlernen von 8-Bit-Mikrocontrollern,
- Arbeit mit einem Mikrocontrollerentwicklungssystem,
- Eingabe und Editieren von Quellprogrammen in der Programmiersprache C,
- Compilieren und Linken von Anwenderprogrammen,
- Download von Anwenderprogrammen in die Zielhardware,
- Starten und praktischer Test von Anwenderprogrammen,
- Durchführung von einfachen Ein-Ausgabeoperationen

Versuchsvorbereitung

Informieren Sie sich über die Leistungsmerkmale (Taktfrequenz, Speicherausstattung, Gehäuse des Controllers, Schnittstellen usw.) des verwendeten Mikrocontrollerboards auf der Basis des 8-Bit-Controllers Dallas DS80C320 (http://www.phytec.de/phytec/fuer_8-bit_module/rapid_development_kit_micromodul-8051.html)!

Informieren Sie sich mithilfe der Kurzbedienungsanleitung (https://www.hs-zigr.de/e-technik/stud/Lehrmatr/P_51_000_NEU.pdf) über die Vorgehensweise bei der Programmentwicklung in der Programmiersprache C mithilfe der Programmieroberfläche *µVision2* der Firma *KEIL*!

Mithilfe des „High Speed Microcontroller User Guides“, Seite 116-118 (https://www.hs-zigr.de/e-technik/stud/ Lehrmatr/kuehne/hsmicro_ug.pdf) ist der Schaltungsaufbau der digitalen Ein-/Ausgänge der 8-Bit-Ports P1 und P3 zu ermitteln? Wieso können die Ports P1 und P3 ohne eine Zusatzbeschaltung wahlweise als Ein- und Ausgänge verwendet werden?

Für die digitalen Ein-/Ausgänge ist ein elektrisches Ersatzschaltbild zu entwerfen!

Worin besteht das Wesen von Strukturen und Units (Varianten) in der Programmiersprache C? Warum ist es günstig, die Eingabebits P3.2 ... P3.5 und die Ausgabebits P1.0 ... P1.7 zu einer Struktur zusammenzufassen und nachfolgend ein Zuweisen dieser Struktur auf eine Unit durchzuführen?

Informieren Sie sich mithilfe der Schaltkreisdokumentation des Mikrocontrollers DS80C320 (http://www.hs-zigr.de/e-technik/stud/Lehrmatr/kuehne/Mikrorechentechnik/ds80c320.pdf) über dessen Pin-Layout, insbesondere über die angebotenen Gehäusevarianten, die zur Verfügung stehenden Ports, sowie der Bedeutung 40 Anschlusspins!

Informieren Sie sich mithilfe des „High Speed Microcontroller User Guides“, Seite 6 (https://www.hs-zigr.de/e-technik/stud/Lehrmatr/kuehne/hsmicro_ug.pdf und der Schaltkreisdokumentation des Mikrocontrollers DS80C320 Seite 1 (http://www.hs-zigr.de/e-technik/stud/Lehrmatr/kuehne/Mikrorechentechnik/ds80c320.pdf über die Hardwareressourcen des DS80C320!

Welche charakteristischen Merkmale besitzt die im Praktikum verwendete Controllervariante DS80C320 QCG!

Was verstehen Sie unter der Standardfunktion und der alternativen Funktion eines I/O-Ports bzw. eine I/O-Pins eines Mikrocontrollers?

Erläutern Sie die Struktur der Speicherkonfikuration, insbesondere der verwendeten Signalleitungen, eines Mikrocontrollersystems mit einem Programmspeicher von 32 kByte und einem Datenspeicher von 2 kByte entsprechend Seite 10, Fig. 3 der Schaltkreisdokumentation des Mikrocontrollers DS80C320 (http://www.hs-zigr.de/e-technik/ stud/Lehrmatr/kuehne/Mikrorechentechnik/ds80c320.pdf)!

Versuchsaufbau

Die 4 binären Eingänge werden mit 4 Umschaltern (a, b, c und d) verbunden, die 8 binären Ausgänge steuern über zusätzliche Treiber 8 Leuchtdioden (a, b, c, d, e, f, und e) an, vgl. Abb. 1. Die gemeinsame Anode der Leuchtdioden befindet sich auf Betriebsspannungspotential von U_B = 5V. Durch eine zusätzliche treiberinterne Invertierung des Eingangssignals werden die Leuchtdioden jedoch bei einem H-Pegel an den digitalen Ausgängen aktiviert.

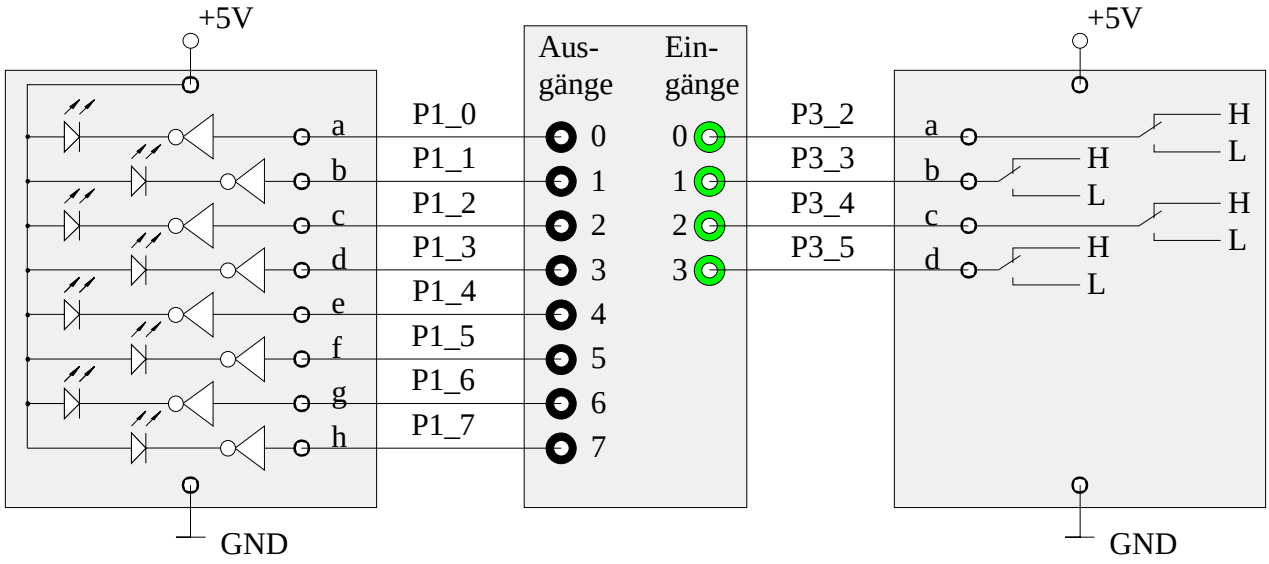

Abb.1: Verdrahtung der Aus- und Eingänge

Aufgabenstellung:

Es ist ein Programm zu entwickeln, welches den Zustand der 4 Eingangsbits P3.2 ... P3.5 erfasst und auf die 4 niederwertigen Ausgangsbits P1.0 ... P1.3 der insgesamt 8 Ausgabebits P1.0 ... P1.7 überträgt! Die restlichen Ausgabebits sollen mit dem Wert Null beschrieben werden! Die Werte der binären Eingabeinformationen sind mithilfe von Schaltern zu variieren und das Ergebnis durch eine Kontrolle der Zustände der Leichtdioden zu überprüfen. Bei der Erstellung des Programms ist insbesondere darauf zu achten, dass nicht benötigte Bits innerhalb von Strukturen, Bytes, Units usw. fest auf 0 und somit auf Low-Potenzial zu setzen sind.
Der Zustand der binären Eingänge ist weiterhin in einer geeigneten Form für jedes Bit einzeln auf dem Monitor mithilfe eines printf-Befehls anzuzeigen. Für die zyklische Abfrage der binären Eingänge, der Übertragung dieser Informationen auf die binären Ausgänge und der Anzeige auf dem Monitor ist eine Endloschleife zu programmieren, die ohne zeitliche Vorgaben zyklisch durchlaufen wird.

Hinweise zur Erstellung des Programms:

- Vor dem Aufruf der Programmieroberfläche µVision2 sind alle Dateien der Directory C:\Programme\phybasic\um8051\DEMOS\Keil\Versuch_1 in eine noch zu erstellende Directory C:\Programme\phybasic\um8051\DEMOS\ Keil\KK_I_P zu kopieren (KK ist dabei die Nummer des Matrikels, I die der Versuchsgruppe und P die Versuchsnummer). Die kopierten Dateien gehören zu einem bereits vordefinierten Projekt, welches unter anderem die Quellcodedatei Ein_Ausgabe.c besitzt. In diese Datei ist der vorbereitete Anwendercode hinzuzufügen. Nach einem Start der Programmieroberfläche µVision 2 erfolgt das Öffnen des Projektes mithilfe von „Project - OpenProject“ und ein Laden der Datei Versuch_1.uv2 aus der Directory C:\Programme\phybasic\um8051\DEMOS\Keil\KK_I_P!

  Die zu vervollständigenden Projektdateien sind auch im Internet in der Datei Ein-Ausgabe.zip verfügbar!

- Durch ein Aktivieren der Datei Ein_Ausgabe.c aus der Liste der Projektdateien per Mausklick im Dateifenster von µVision2 wird diese automatisch in den Editor geladen!

- Der Programmcode ist in eine Endlosschleife einzufügen und wird dort zyklisch, entsprechend der Laufzeit des Programms wiederholt abgearbeitet
  (vgl. Seite 1 https://www.hs-zigr.de/e-technik/stud/Lehrmatr/ P_51_000_NEU.pdf)!

- Die Zustände der 4 binären Eingänge sind für jedes Bit einzeln bei jedem Schleifenfurchlauf in geeigneter Form für jeden Eingang durch die Ausgabe einer „0“ oder einer „1“ mithilfe des printf-Befehls und einer entsprechenden Formatierung auf dem Bildschirm anzuzeigen!

- Für Wertzuweisungen auf die Ausgabeports P1_0 ... P1_7 sind wenn möglich die im Quellkode bereits vordeklarierten und vordefinierten Strukturen und Units zu verwenden.

Die Programmieraufgabe ist vollständig umgesetzt, wenn beliebig eingestellte Eingangskombinationen der 4 Eingabebits fehlerfrei auf die 4 niederwertigsten Ausgabebits übertragen werden, die 4 höchstwertigsten Ausgabebits ständig Low-Pegel führen und die Zustände der binären Eingänge gleichzeitig für jedes Bit einzeln auf dem Monitor dargestellt werden.

Hinweis:
Bei den Internetadressen, die auf Ziele außerhalb der Hochschule verweisen kann es sein, dass diese unter Umständen nicht mehr gültig sind. Dann ist unter Verwendung der in der Regel unveränderten Domain-Adresse die gesuchte Internetseite selbständig aufzufinden!

Listing des Programms Ein_Ausgabe.c

```
#include <reg320.h>          /* include 8051 header file        */
#include <stdio.h>           /* Standard I/O functions          */

sbit P1_0 = P1^0;
sbit P1_1 = P1^1;
sbit P1_2 = P1^2;
sbit P1_3 = P1^3;
sbit P1_4 = P1^4;
sbit P1_5 = P1^5;
sbit P1_6 = P1^6;
sbit P1_7 = P1^7;

sbit P3_2 = P3^2;
sbit P3_3 = P3^3;
sbit P3_4 = P3^4;
sbit P3_5 = P3^5;

/*Deklaration einer Struktur für die Eingabebits*/

struct bitfeld_in
      {
      unsigned bit_in_nr0 :1;
      unsigned bit_in_nr1 :1;
      unsigned bit_in_nr2 :1;
      unsigned bit_in_nr3 :1;
      unsigned dummy :12;
       }flags_in;

/*Deklaration einer Struktur für die Ausgabebits*/
struct bitfeld_out
      {
     unsigned bit_out_nr0 :1;
     unsigned bit_out_nr1 :1;
     unsigned bit_out_nr2 :1;
     unsigned bit_out_nr3 :1;
     unsigned bit_out_nr4 :1;
     unsigned bit_out_nr5 :1;
     unsigned bit_out_nr6 :1;
     unsigned bit_out_nr7 :1;
     unsigned dummy :8;
     }flags_out;

/*Deklaration einer union für die Eingabebits */
union flagss_in
      {
      struct bitfeld_in bits_in;
      unsigned int bit_wert_in_uint;
      char bit_in[2];
      }input_bit;
```

```
/*Deklaration einer union für die Ausgabebits */
union flagss_out
      {
      struct bitfeld_out bits_out;
      unsigned int bit_wert_out_uint;
      char bit_out[2];
      }output_bit;

char getkeyy()
     {
     char c;
     while (!RI);
     while (SBUF==17) RI=0;
     c = SBUF;
     RI = 0;
     return (c);
     }

void init()
    {
    TI=1;
    ES0=0; //SerialInterrupt Disabled
    }

void control()
    {
    if (SBUF0==0x1b)                    //Stop bei ESC
        {
        printf("\n SYSTEM GESTOPPT");
        ES0=1;                          //SerialInterrupt Enabled
        while(1);                       //RESET um neu zu beginnen
        }
        RI=0;
    }

void main (void)
    {
    int i;
    init();
    …..
    while (1)   /* loop forever                */
        {

/*************************************************************************
 ********** hier den eigenen Programmkode einfügen      ********************
 *************************************************************************/
        control();
        printf("%i% \r",i++);
        }
    }
```