Dokument NR.: I2C-004-U-02

8-BIT ADC UNDER UND OVER RANGE

ADC mit ADC081C021 „under range“ und „over range“ Register (I2C-004 Board)

Inhaltsverzeichnis

1. Aufgabevon.....3

1.1 Verwendete Module: 3

1.2 „Target Mission“.....3

2. Lösung.....4

Modification History:

Version	Date	Comments
1.0	03.2013	first release

1. Aufgabevon

- Initialisieren Sie ADC081C021 über I²C Schnittstelle
  - Die I²C Initialisierung-, Sende- und Empfang-Funktionen werden in diesem Beispiel vorausgesetzt
- Initialisieren Sie Interrupt von ADC081C021
- Legen Sie Spannungsgrenzen fest
  - untere Grenze 1000mV obere grenze 3000mV
  - so bald die untere oder obere Grenze unter- bzw. überschritten ist wird ein Interrupt ausgelöst
  - bei Interrupt leuchtet blaue LED P1 auf
- Lesen Sie den Wert aus dem „Conversation Result Register" aus
- Berechnen Sie die dezimalen Spannungswerte
- Geben Sie die gemessene Spannungswerte via UART auf Hyperterminal aus

1.1 Verwendete Module:

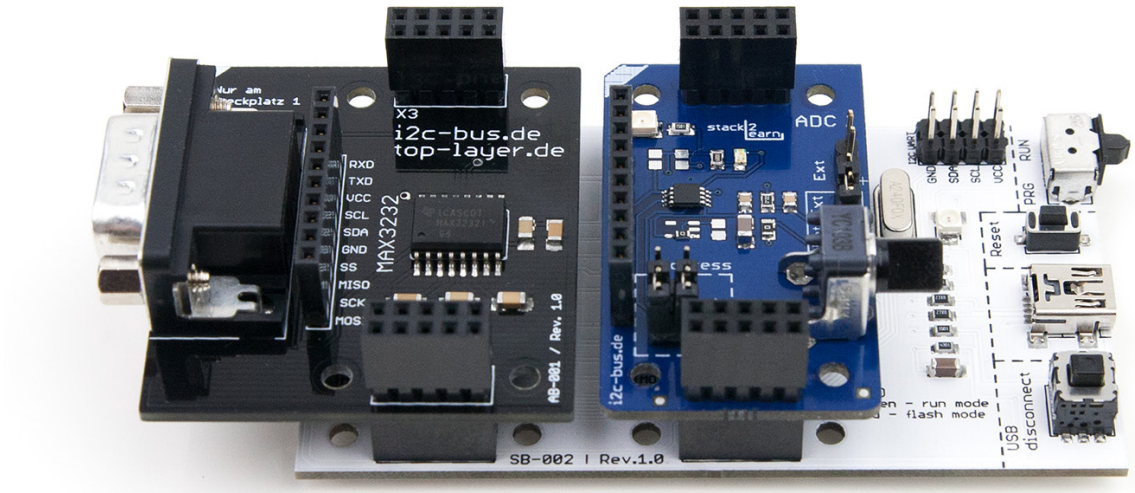
8051 Mikrocontroller Board mit AT89C5131A-RDTUM 24MHz Quarz (SB-001/SB-002), ADC Board (I2C-004) und RS232 Modul (AB-001/AB-002).

1.2 „Target Mission"

Ein Video sagt mehr, als tausend geschriebene Worte.

In diesem Video wird gezeigt wie das Ergebnis aussehen soll (I2C-004-U-02). Es wird kein Quellcode gezeigt oder erklärt.

„Target Mission" auf youTube



2. Lösung

```
/*****************************************************************************
   Datei:     I2C-004-U-02-adc-under-over-limit.c
   Produkt:   I2C-004-U-02
   Projekt:   ADC081C021 Initialisierung

   Datum:     Mar.2013 - first release
   Version    1.0
   Kompiler:  Keil V9.05

   Autor:     Viktor Schabelski info@i2c-bus.de.de
   Lizenz:    Creative Commons Lizenz
              Namensnennung - Keine kommerzielle Nutzung - Keine Bearbeitungen
              www.i2c-bus.de/cc

   - Initialisieren Sie ADC081C021 ueber I2C Schnittstelle
      -  Die eigenen I2C Initialisierung-, Sende- und Empfangfunktionen werden in diesem
         Beispiel vorausgesetzt
   - Initialisieren Sie Interrupt von ADC081C021
   - Legen Sie Spannungsgrenzen fest
      -  untere Grenze 1000mV obere grenze 3000mV
      -  so bald die untere oder obere Grenze unter- bzw. überschritten ist,
         wird ein Interrupt ausgelöst
      - bei Interrupt leuchtet blaue LED P1 auf
   - Lesen Sie den Wert aus dem „Conversation Result Register“ aus
   - Berechnen Sie die dezimalen Spannungswerte
   - Geben Sie die gemessene Spannungswerte via UART auf Hyperterminal aus

*****************************************************/

/*****************************************************************************
   Includes
*****************************************************/
#include <at89c5131.h>
#include <emuTWI.h>


/*Dualnotation 8 Stellen
// Implementierung P3 = B8(0, 1, 1, 0, 1, 0, 1, 1);*/
#define __B8(a, b, c, d, e, f, g, h)\
     (a<<7|b<<6|c<<5|d<<4|e<<3|f<<2|g<<1|h<<0)


#define ADC_ADRESS    0xAA /* default value: conect ADR0 and ADR1 to gnd => 0xAA*/

/* Register */
#define ADC_RESULT          0x00
#define ADC_CONFIG          0x02
#define ADC_UNDER_RANGE     0x03
#define ADC_OVER_RANGE      0x04



void ausgabe (unsigned char *);
void uart_init(unsigned char );
void uart_sendChar(unsigned char );
void adc8_setLimit (unsigned char , unsigned int );
```

```
/*****************************************************************************
   Funktionen
*****************************************************************************/
/*===========================================================================
*  name    :  main
*  input   :  none
*  output  :  none
*  descr.  :  Main Funktion
-----------------------------------------------------------*/
void main(void){
   unsigned char aData[5] ={0x00};

   uart_init(24);

/* ADC Initialisierung */
   aData[0] = ADC_ADRESS;              // Slave Adress
   aData[1] = ADC_CONFIG;
   aData[2] = __B8(0,0,1,0,0,1,0,0);    // Automatic Conversion mode Tconvert x 32
   emuTWI_sendData (aData, 3);          // Daten senden via I2C

   adc8_setLimit(ADC_UNDER_RANGE, 1000);
   adc8_setLimit(ADC_OVER_RANGE, 3000);


   while(1){
      aData[0] = ADC_ADRESS;
      aData[1] = ADC_RESULT;
      emuTWI_sendData(aData, 2);

   /* Daten empfangen */
      aData[0] = ADC_ADRESS + 1;   // Slave Adresse + RW
      emuTWI_receiveData(aData, 2);

      ausgabe(aData);
   }
}

/*===========================================================================
*  name :  ausgabe
*  input:  Array mit 2 Bytes
*  output  :  none
*  descr.  :  Ausgabe von Messwerten via UART auf Hypeterminal
-----------------------------------------------------------*/
void ausgabe (unsigned char *pData){
   unsigned char MSB = 0x00, LSB = 0x00;
   unsigned int uiErgebnis = 0;

      MSB = (pData[0] << 4) & 0xF0;
      LSB = (pData[1] >> 4) & 0x0F;

      uiErgebnis = (MSB|LSB);
      uiErgebnis *= 20;       //ungefähr (5000mV / 256)

      uart_sendChar (((uiErgebnis/1000)%10)+48);   // Tausender
      uart_sendChar(,,');
      uart_sendChar (((uiErgebnis/100)%10)+48); // Hunderter
      uart_sendChar (((uiErgebnis/10)%10)+48);  // Zehner
      uart_sendChar ((uiErgebnis%10)+48);       // Einer
      uart_sendChar (, ,);
      uart_sendChar (,m');
      uart_sendChar (,V');
      uart_sendChar (0x0D);
}
```

```
/*=========================================================================
*  name  : uart_init
*  input : uchQuarz - Quarzfrequenz - erlaubte Werte 12 fuer 12MHz und 24 fuer 24MHz
*  output  : none
*  descr. :   Initialisierung RS232/V24 Schnittstelle.
*       Initialisierung der seriellen Schnittstelle in Mode 1
*       Schnittstellenparameter: 9600Baud, 8 Datenbit, 1 Stopp-Bit, asynchroner Betrieb mit
Empfang
*       Quarz Frequenz 12/24 MHz
-------------------------------------------------------*/
void uart_init(unsigned char uchQuarz){
// Schnittstelle Initialisierung
   SCON |= 0x50; // SM1 (2^6) = 1; REN (2^4) = 1;
//-----------------------------------------------
// Timer2 Initialisierung
   T2CON |= 0x30; // TCLK (2^4) = 1; RCLK (2^5) = 1;
   TR2 = 1; // Timmer2 run T2CON 2^2
   RCAP2H = 0xFF;
   if(uchQuarz == 12) RCAP2L = 0xD9; //Reloadwert bei 12MHz Quarz
   else RCAP2L = 0xB2;               //Reloadwert bei 24MHz Quarz
   TI = 1;
}


/*=========================================================================
*  name  :  uart_sendChar
*  input :  uchCharacter - Zeichen zum Senden
*  output  :  none
*  descr.  :  Sende Zeichen ueber RS232/V24 Schnittstelle.
-------------------------------------------------------*/
void uart_sendChar(unsigned char uchCharacter){
   SBUF = uchCharacter; /* sende Zeichen */
   while(TI == 0);      /* warte bis Zeichen gesendet ist */
   TI = 0;
}


/*=========================================================================
*  name  :  adc8_setLimit
*  input :  uchLimitRegister - Register Pointer
*  output  :  uiLowLimitVoltage: Spannungswert in mV
*  descr.  :  schreibe in „under range“ oder „over range“ Register
-------------------------------------------------------*/
void adc8_setLimit (unsigned char uchLimitRegister, unsigned int uiLowLimitVoltage){
   unsigned char aData[4] = {0x00};

   uiLowLimitVoltage /= 20;     // ((ADC_8_REFERENCE_U/ADC_8_RESOLUTION)+1

   aData[0] = ADC_ADRESS;
   aData[1] = uchLimitRegister;
   aData[2] = (unsigned char)((uiLowLimitVoltage & 0x00F0) >> 4);
   aData[3] = (unsigned char)((uiLowLimitVoltage & 0x000F) << 4);

   emuTWI_sendData(aData, 4);
}
```

Haben Sie einen Fehler entdeckt?

Wir sind dankbar für Ihren Hinweis. Schicken Sie uns bitte diesen Hinweis einfach per E-Mail: info@i2c-bus.de.

Vielen Dank!

Dok. NR.: I2C-004-U-02