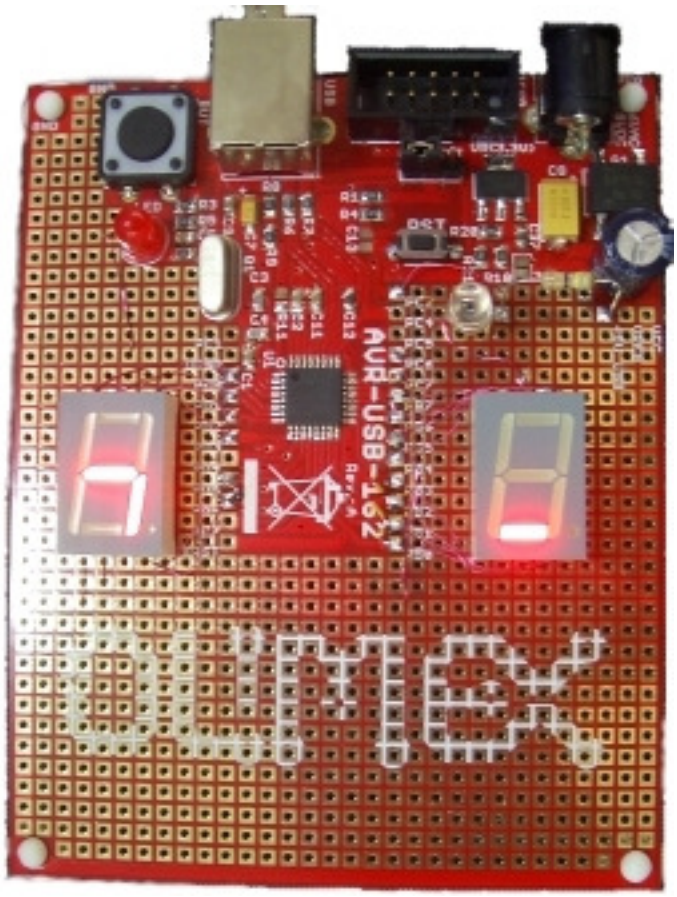
Projektarbeit in DVT
auf dem Entwicklerboard
OUMEX AVR-USB-162

von

Michael Eggl , Markus Vogl



1 FROM 8
Version 1.0

Inhaltsverzeichnis:

Aufgabenstellung Seite 3
Planung Seite 3
Quellcode Seite 3- 6
Beschreibung der Funktionen Seite 7

- main
- dg1
- dg2
- check button
- entprellen
- ende

Aufgabenstellung:

Programmierung eines Spiels, das die auf dem Entwicklerboard vorhandenen Möglichkeiten nutzt.

Planung:

Nutzung der beiden auf dem Board befindlichen 7-Segment-Anzeigen zur Visualisierung des Spiels.
Zu Beginn des Spiels sollen auf der rechten 7-Segment-Anzeige mit einer Verzögerung von 100ms die Leuchtdioden a-f nacheinander in Form einer 8 aufleuchten.
Der Benutzer muss nun durch den auf dem Board befindlichen Taster eine der 7 Leuchtdioden (a-g) auswählen.
Nach der Auswahl beginnt nun die linke 7-Segment-Anzeigen zu laufen.
Durch drücken des Tasters muss nun auf der linken Anzeige dieselbe Leuchtdiode „getroffen" werden, wie zuvor auf der rechten Anzeige.
Wenn dies der Fall ist, wird die Verzögerungszeit um 20ms verkleinert und das Spiel beginnt von neuem.
Das ganze wiederholt sich solange bis eine Verzögerungszeit von 0ms erreicht wird.
Danach kann das Spiel von Neuem gestartet werden.

Quellcode:

```
#include<avr/io.h>
#define F_CPU 1000000
#include <util/delay.h>
#define DELAY 500                       //Anzeigedauer der Buchstaben
#define ANZAHL_STABIL 1000

void dg1(int);
void dg2(int taste1,int max);
void entprellen(void);
void check_button(void);
void ende(void);

//void taster(void);
```

Funktion main

```
int main(void)
{
        DDRB=0xff;              //PORTB als AUSGÄNGE setzen
        DDRC=0xC0;              //PORTD6 und 7 als AUSGANG setzen
        DDRD=0x6F;              //PORTD 0,1,2,3,5,6 als AUSGANG setzen
        PORTD=0x80;             //PORTD7 auf HIGH setzen

        int max=100;            //festlegen der Verzögerungszeit
        dg1(max);               //Aufrufen der Funktion dg1 mit Übergabewert max

}
```

Funktion dg1

```
void dg1(int max)
{
	//                          a     b     g     e     d     c     g     f
	unsigned char anzr[] = {0x20,0x10,0x80,0x01,0x02,0x04,0x80,0x40};
	int taste1=8;
	int i;

		if(max<=0)
		{
			ende();
		}
		else
		{

			while(taste1==8)
			{
				for(i=0;i<=7;i++)
				{
					PORTB=anzr[i];


					if(!(PIND &(1<<PIND7)))
					{
						check_button();
						taste1=i;
						dg2(taste1,max);
					}
					_delay_ms(max);		//Geschwindigkeit der LED

				}
			}
		}
}
```

Funktion dg2

```
void dg2(int taste1,int max)
{
        int c,j;
        unsigned char anzl[8][2]=  {  {0x04,0x00},          //a
                                      {0x08,0x00},          //b
                                      {0x01,0x00},          //g
                                      {0x00,0x40},          //e
                                      {0x00,0x80},          //d
                                      {0x40,0x00},          //c
                                      {0x01,0x00},          //g
                                      {0x02,0x00}};         //f

                while(1)
                {
                        PORTD=0x80;

                        for(j=0;j<=7;j++)
                        {
                                PORTD=anzl[j][0];
                                PORTC=anzl[j][1];
                                _delay_ms(max);

                                if(!(PIND &(1<<PIND7)))
                                {
                                        check_button();

                                        if(j==taste1)           //Anzeige ob richtig
                                        {
                                                for(c=0;c<=3;c++)
                                                {
                                                        PORTD=0x4F;
                                                        PORTC=0xc0;
                                                        PORTB=0x00;
                                                        _delay_ms(DELAY);
                                                        PORTD=0x00;
                                                        PORTC=0x00;
                                                        PORTB=0xf7;
                                                        _delay_ms(DELAY);
                                                }
                                                max=max-20;         //Geschwindigkeit erhöhen
                                                dg1(max);
                                        }
                                }
                        }
                }
}
```

Funktion entprellen

```
void entprellen()
{
	int i=0;
	int current_state,last_state =0;
	while(i<ANZAHL_STABIL)
	{
		i++;
		current_state=(PIND&(1<<7));
		if(current_state!=last_state)
		{
			i=0;
			last_state = current_state;
		}
	}
}
```

Funktion check button

```
void check_button()
{
	if(PIND&(1<<7))
	{
		return;
	}
	entprellen();
	while((PIND&(1<<7))==0);
	entprellen();
}
```

Funktion ende

```
void ende(void)
{
	int i;

	//                          E      n      d     E
	unsigned char anzr[] = {0xE3,0x85,0x97,0xE3};

		while(1)
		{
			for(i=0;i<=3;i++)
			{
				PORTB=anzr[i];
				_delay_ms(DELAY);          //Anzeigedauer der Buchstaben
			}
			PORTB=0x00;
			_delay_ms(DELAY);
		}
}
```

Quellcode Beschreibung:

main():
Die Funktion main dient dazu, die benötigten Ports als Ausgang oder Eingang zu beschalten. Des Weiteren wird hier die Verzögerungszeit für den Beginn des Spiels festgelegt.
Im Anschluss wird nun die Funktion dg1(Durchgang1) mit dem Übergabewert max aufgerufen.

dg1():
In der Funktion dg1 befindet sich ein eindimensionales unsigned char Array welches mit den Hexwerten, die für die Ansteuerung der LEDs von a-g der rechten Anzeige benötigt werden, befüllt ist.
Des Weiteren werden hier noch die Integer-Variablen „taste1“ und „i“ eingeführt.
Die Variable taste1 wird am Anfang mit dem Wert 8 belegt.
Am Anfang wird durch eine if-Schleife abgefragt, ob der Wert max kleiner oder gleich 0 ist, sollte dies der Fall sein wird die Funktion „ende“ aufgerufen.
Wenn max größer 0 ist fährt das Programm im else-Zweig weiter.
Hier wird eine Endlosschleife aufgerufen, die auf der 7-Segment-Anzeige nacheinander alle Leuchtdioden in Form einer 8 ansteuert.
Wenn nun der auf dem Bord befindliche Taster gedrückt wird, wird die Funktion check_button aufgerufen und der Variable „taste1“ der Wert der leuchtenden LED zugewiesen.

dg2():
In der Funktion dg2 befindet sich ein zweidimensionales unsigned char Array welches mit den Hexwerten, die für die Ansteuerung der LEDs von a-g der linken Anzeige benötigt werden, befüllt ist. In der Endlosschleife while(1) wird als erstes die for-Schleife aufgerufen um das Lauflicht in der linken Anzeige zu starten. Bei jedem Durchgang wird mit der Abfrage „if(!(PIND &(1<<PIND7)))“ geprüft ob der Taster erneut gedrückt wurde. Bei gedrücktem Taster wird erst die Entprellfunktion „check_button()“ aufgerufen und dann mit „if(j==taste1)“ geprüft ob die aktuelle Position der linken 7-Segment-Anzeige mit der der rechten übereinstimmt. Falls das zutrifft wird in jeder Anzeige abwechselnd eine 8 dargestellt (4x) und ein neuer Durchgang mit erhöhter Geschwindigkeit aufgerufen.

check button() und entprellen():
Mit der Funktion „check_button“ und „entprellen“ wird der Taster auf einen festen Zustand hin überprüft (geöffnet/geschlossen). So wird sichergestellt, dass das Prellen des Tasters eliminiert wird.

ende():
Hier wird in der rechten Anzeige das Wort „E n d e“ dargestellt und das Spiel muss neu gestartet werden.