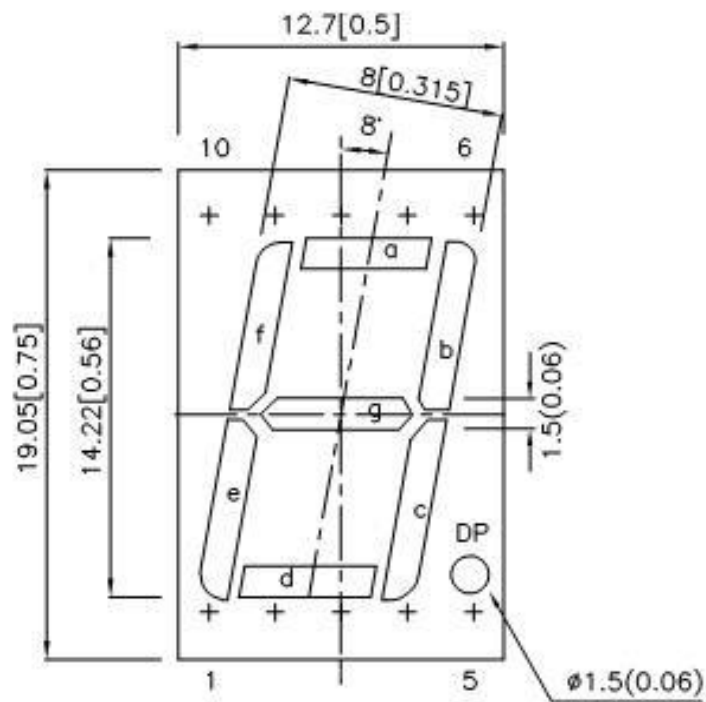


Geschicklichkeitsspiel mit dem AT90 USB162

Projektteam : Lennard Main
Christian Kahl

Hardware: AT90USB162 auf Entwicklerboard von OLIMEX mit zwei 7-Segment Anzeigen.
Die Programmierung sowie die Stromversorgung erfolgt über den USB-Anschluss.

Die 7-Segment Anzeigen stammen von der Firma Kingbright und sind wie folgt belegt:



Anzeige links:

PIN a: PORTD2
PIN b: PORTD3
PIN c: PORTD6
PIN d: PORTC6
PIN e: PORTC7
PIN f: PORTD0
PIN g: PORTD1

Anzeige rechts:

PIN a: PORTB5
PIN b: PORTB4
PIN c: PORTB2
PIN d: PORTB1
PIN e: PORTB0
PIN f: PORTB6
PIN g: PORTB7

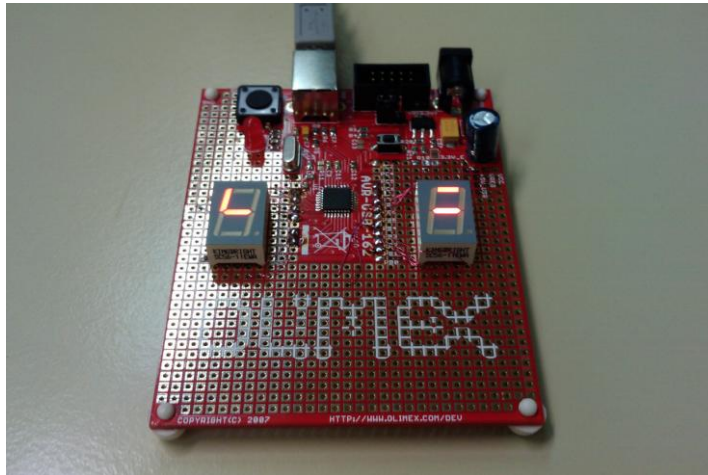
Das Spiel:

Ziel ist es, durch Drücken des Tasters die linke Anzeige mit nur 10 Versuchen aufzufüllen.

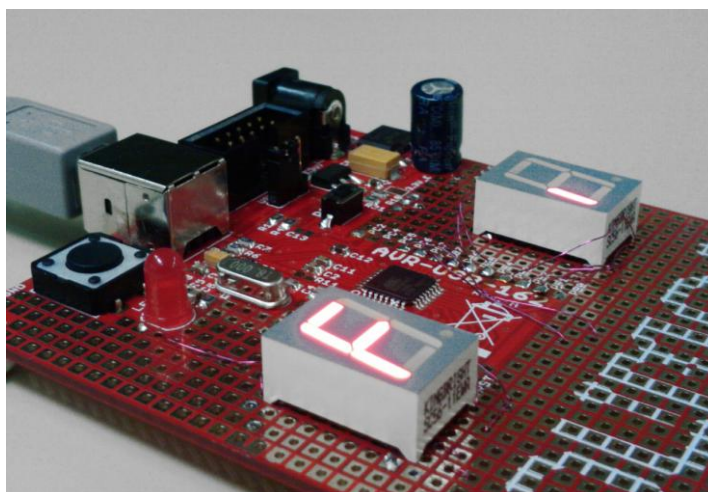
Realisierung:

Die rechte 7 - Segmentanzeige ändert in einem Takt von 1000 ms ihren Zustand, und lässt in der Reihenfolge a,b,c,d,e,f,g alle Segmente der Anzeige blinken.

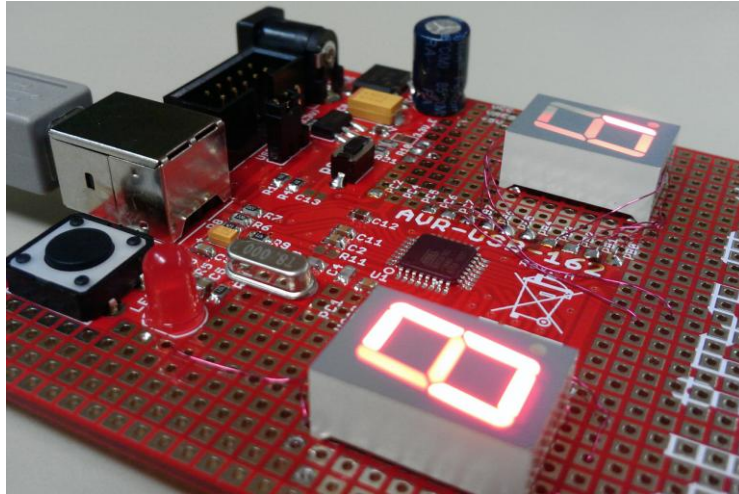
Durch Drücken des Tasters wird das entsprechende Element auf der linken Anzeige aktiviert.



Werden mehr als 10 Versuche benötigt um die Anzeige aufzufüllen, ist das Spiel verloren und in der linken Anzeige wird der Buchstabe „F“ angezeigt.



Ist die Anzeige nach weniger als 10 Versuchen gefüllt, so beginnen alle Segmente der rechten Anzeige zu blinken.



Quellcode:

Der Quellcode wurde in der Programmiersprache C verfasst, mit Hilfe des avr-gcc kompiliert und mit „FLIP“ auf den Mikro - Controller übertragen.

Hier der Quellcode:

```
#include <avr/io.h>
#include <avr/interrupt.h>
#define F_CPU 100000
#include <util/delay.h>
#include <avr/wdt.h>

void anzeige(int); // Definition der Funktionen
void error ();

volatile int anz=0; // Definition der Variablen
volatile int versuch =0;
volatile int check =0;

int main (void)
{
```

```

int blink=1, b1, b2, b3, b4, b5, b6, b7; //Reihenfolge der einzelnen Segmente
DDRB =0xff; // Register fuer rechte 7 Segment Anzeige
DDRD |= (1<<PORTD2); //a // Register fuer linke 7 Segment Anzeige
DDRD |= (1<<PORTD3); //b
DDRD |= (1<<PORTD6); //c
DDRC |= (1<<PORTC6); //d
DDRC |= (1<<PORTC7); //e
DDRD |= (1<<PORTD0); //f
DDRD |= (1<<PORTD1); //g
DDRD |= (1<<PORTD5);

PORTD &= ~(1<<PORTD7); // Eingang fuer den Taster
while(1)
{
    switch (blink){ // Unterscheidung der verschiedenen Zustaeude
        case 1: PORTB = 0x20;
                b1=0x20;
                if (!(PIND & (1<<PORTD7)))
                {
                    _delay_ms(50);
                    anzeige(b1);
                }break;
        case 2: PORTB = 0x10;
                b2= 0x10;
                if (!(PIND & (1<<PORTD7)))
                {
                    _delay_ms(50);
                    anzeige(b2);
                }break;
        case 3: PORTB = 0x04;
                b3= 0x04;
                if (!(PIND & (1<<PORTD7)))
                {
                    _delay_ms(50);
                    anzeige(b3);
                }break;
        case 4: PORTB = 0x02;
                b4=0x02;
                if (!(PIND & (1<<PORTD7)))
                {
                    _delay_ms(50);
                    anzeige(b4);
                }break;
    }
}

```

```

case 5: PORTB = 0x01;
        b5=0x01;
        if (!(PIND & (1<<PORTD7)))
        {
            _delay_ms(50);
            anzeige(b5);
        }
        break;

```

```

case 6: PORTB = 0x40;
        b6=0x40;
        if (!(PIND & (1<<PORTD7)))
        {
            _delay_ms(50);
            anzeige(b6);
        }break;

```

```

case 7: PORTB = 0x80;
        b7=0x80;
        if (!(PIND & (1<<PORTD7)))
        {
            _delay_ms(50);
            anzeige(b7);
        }break;

```

```
};
```

```
_delay_ms(1000); // Zeit wie lange ein einzelnes Segment blinkt
```

```
blink++;
```

```
if (blink ==8) // ist 8 erreicht, wird wieder bei 1 angefangen
```

```
{
    blink=1;
}
```

```
if (versuch >=10)
```

```
{
    error();
}
```

```
if (PIND &(1<<PORTD0)&&(1<<PORTD1)&&(1<<PORTD2)&&(1<<PORTD3)&&(1<<PORTD6))
```

```
{
    if (PINC & (1<<PORTC6)&&(1<<PORTC7))
```

```
    PORTB =0xff; //lass die Punkte blinken....
```

```
    _delay_ms(1000);
```

```
}
```

```
}
```

```
}
```

```
void anzeige(int i) // Funktion zum Anzeigen der bereits gelösten Segmente
```

```
{  
    if (i== 0x20)  
    {        PORTD |= (1<<PORTD2);  
        versuch ++;  
    }  
    else if (i== 0x10)  
    {        PORTD |= (1<<PORTD3);  
        versuch ++;  
    }  
    else if (i== 0x04)  
    {        PORTD |= (1<<PORTD6);  
        versuch ++;  
    }  
    else if (i== 0x02)  
    {        PORTC |= (1<<PORTC7);  
        versuch ++;  
    }  
    else if (i== 0x01)  
    {        PORTC |= (1<<PORTC6);  
        versuch ++;  
    }  
    else if (i== 0x80)  
    {        PORTD |= (1<<PORTD0);  
        versuch ++;  
    }  
    else if (i== 0x40)  
    {        PORTD |= (1<<PORTD1);  
        versuch ++;  
    }  
}
```

```

void error () // Funktion, die „F“ in der linken Anzeige ausgibt, falls mehr als 10 Versuche verwendet wurden
{
    while (1)
    {

        PORTD &= ~(1<<PORTD3); //b // Deaktivieren der nicht benötigten Ports
        PORTD &= ~(1<<PORTD6); //c
        PORTC &= ~(1<<PORTC7); //d

        PORTD |= (1<<PORTD2); //a
        PORTC |= (1<<PORTC6); //e
        PORTD |= (1<<PORTD0); //f
        PORTD |= (1<<PORTD1); //g
        _delay_ms(500);
        PORTD &= ~(1<<PORTD2); //a
        PORTC &= ~(1<<PORTC6); //e
        PORTD &= ~(1<<PORTD0); //f
        PORTD &= ~(1<<PORTD1); //g
        _delay_ms(500);
        PORTD |= (1<<PORTD2); //a
        PORTC |= (1<<PORTC6); //e
        PORTD |= (1<<PORTD0); //f
        PORTD |= (1<<PORTD1); //g
        _delay_ms(500);
        PORTD &= ~(1<<PORTD2); //a
        PORTC &= ~(1<<PORTC6); //e
        PORTD &= ~(1<<PORTD0); //f
        PORTD &= ~(1<<PORTD1); //g
    }
}

```

*/*Copyright (c) 2010 Lennard Main, Christian Kahl*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright © 2010 Andreas Loy

Hiermit wird unentgeltlich, jeder Person, die eine Kopie der Software und der zugehörigen Dokumentationen erhält, die Erlaubnis erteilt, diese uneingeschränkt zu benutzen, inklusive und ohne Ausnahme, dem Recht, sie zu verwenden, kopieren, ändern, fusionieren, verlegen, verbreiten, unterlizenzieren und/oder zu verkaufen, und Personen, die diese Software erhalten, diese Rechte zu geben, unter den folgenden Bedingungen:

Der obige Urheberrechtsvermerk und dieser Erlaubnisvermerk sind in alle Kopien oder Teilkopien der Software beizulegen.

DIE SOFTWARE WIRD OHNE JEDE AUSDRÜCKLICHE ODER IMPLIZIERTE GARANTIE BEREITGESTELLT, EINSCHLIESSLICH DER GARANTIE ZUR BENUTZUNG FÜR DEN VORGESEHENEN ODER EINEM BESTIMMTEN ZWECK SOWIE JEDLICHER RECHTSVERLETZUNG, JEDOCH NICHT DARAUF BESCHRÄNKT. IN KEINEM FALL SIND DIE AUTOREN ODER COPYRIGHTINHABER FÜR JEDLICHEN SCHADEN ODER SONSTIGE ANSPRÜCHE HAFTBAR ZU MACHEN, OB INFOLGE DER ERFÜLLUNG EINES VERTRAGES, EINES DELIKTES ODER ANDERS IM ZUSAMMENHANG MIT DER SOFTWARE ODER SONSTIGER VERWENDUNG DER SOFTWARE ENTSTANDEN./**