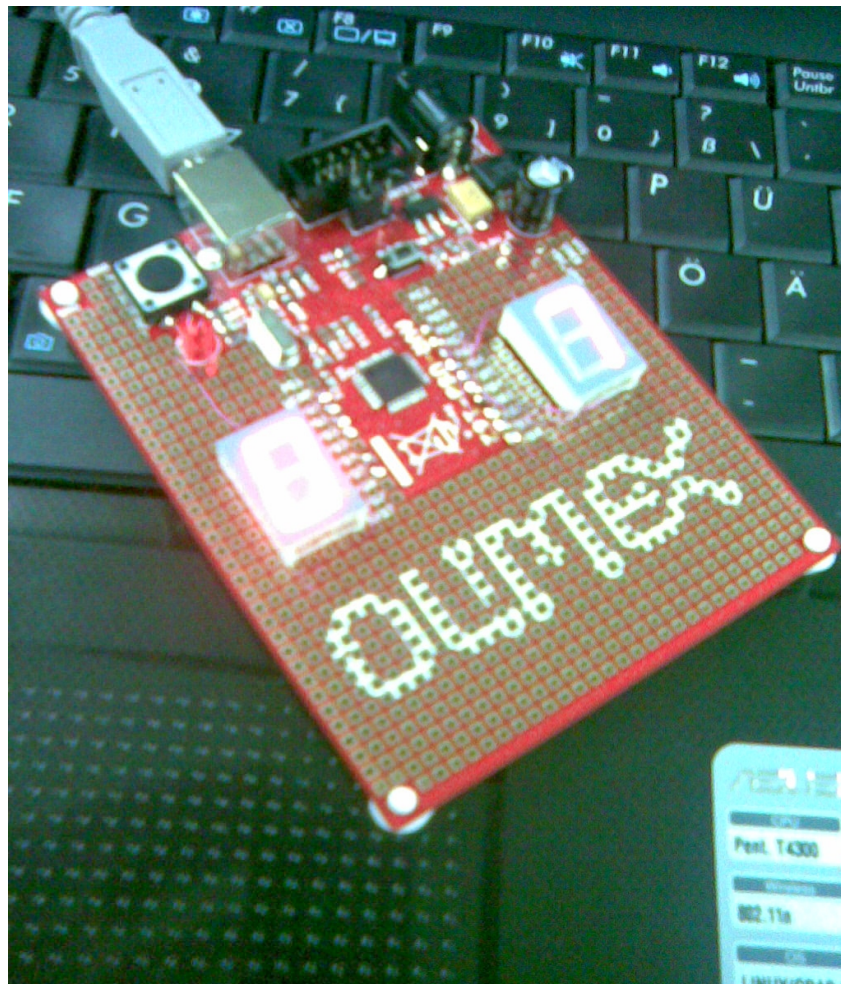


# Projektarbeit

aus der  
Datenverarbeitung

## „Lotto“

von: Hubert Schlenk  
21.07.2010



Olimex AVR USB 162 Entwicklerboard

## **Aufgabenstellung:**

Die Aufgabe war die Entwicklung eines kleinen Programmes / Spieles auf einem Mikrocontroller Entwicklerboard Olimex AVR USB 162.

Weiterhin sollte eine Dokumentation erstellt werden.

Die Projektarbeit wurde während des DVT-Unterrichts durchgeführt.

## **Hardware:**

Olimex AVR USB 162

2 x 7 Segment Anzeige

1 x rote LED

1 x Taster

Der Mikrocontroller AT90USB162 von Atmel bietet 16kB Flash, 512B RAM und 512B EEPROM. Der Chip hat zusätzlich ein USB-Schnittstelle für Programmierung und USART-Kommunikation integriert.

Die Entwicklerplatine AVR-USB-162 von Olimex übernimmt die Spannungsversorgung des Controllers, zusätzlich sind ein Resetbutton, ein Taster, eine LED und ein externer Quarz mit 8MHz im Lieferzustand enthalten.

Die I/O-Pins sind auf Lötunkten herausgeführt und die Platine hat einen Lochrasterbereich, so kann sie leicht mit Bauteilen erweitert werden.

Für dieses Projekt wurden zwei 7-Segmentanzeigen von Kingbright (Abb.1) per Fädeldraht auf Port B, Port C und Port D angelötet, die genaue Belegung der Ports zeigt folgende Tabelle.

## Belegung der Siebensegmentanzeige

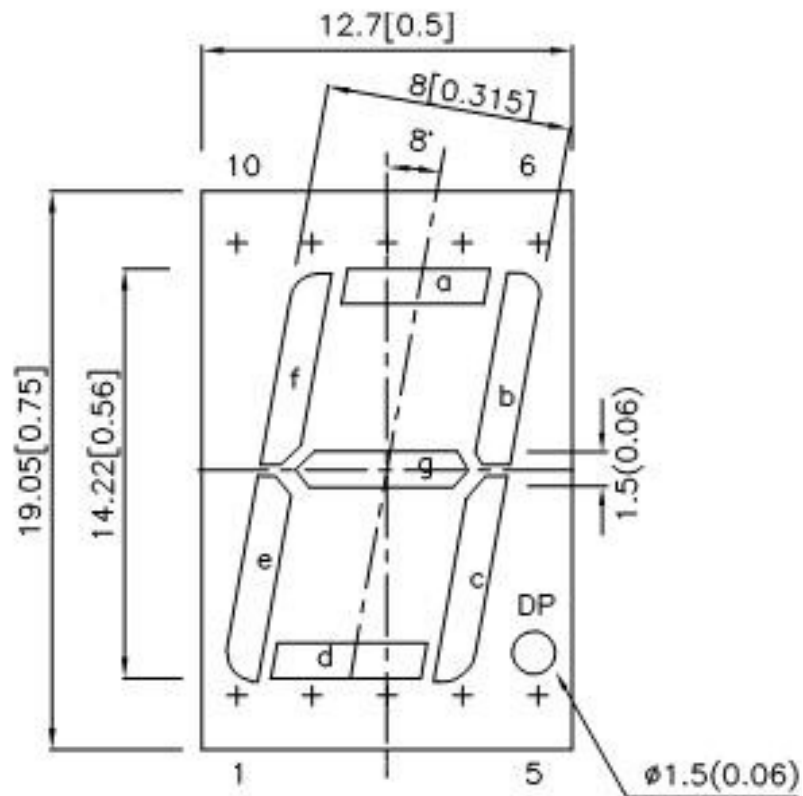


Abb.1

### Rechte Anzeige:

PIN a PORT B5  
PIN b PORT B4  
PIN c PORT B2  
PIN d PORT B1  
PIN e PORT B0  
PIN f PORT B6  
PIN g PORT B7  
PIN DP PORT B3

### Linke Anzeige:

PIN a PORT D2  
PIN b PORT D3  
PIN c PORT D6  
PIN d PORT C7  
PIN e PORT C6  
PIN f PORT D1  
PIN g PORT D0  
PIN DP PORT D5

### LED:

PORT D4

### Taster:

PORT D7

### Software:

AVR-Studio  
Flip 3.4.1

## Spielidee:

Mit Hilfe des Mikrocontrollers kann man 6 Lottozahlen (6 aus 49) ermitteln. Nach der 6. Zahl werden diese nochmal in der gezogenen Reihenfolge angezeigt.

## Ausführung:

Die Zahlen laufen von 00 bis 49 im Millisekunden-Takt. Durch Drücken des Tasters wird der Durchlauf angehalten. Die erste Zahl ist gezogen.

Beim Loslassen des Tasters läuft der Zähler weiter.

Dies wiederholt sich dann 5 mal. Wenn die 00 gezogen wird, wird diese Zahl nochmal ermittelt.

Im Anschluss werden die gezogenen Zahlen nochmals angezeigt.

Die erste Anzeige zeigt die wievielte Zahl dies ist (Bild1) – im Anschluss die gewürfelte Zahl (Bild2).

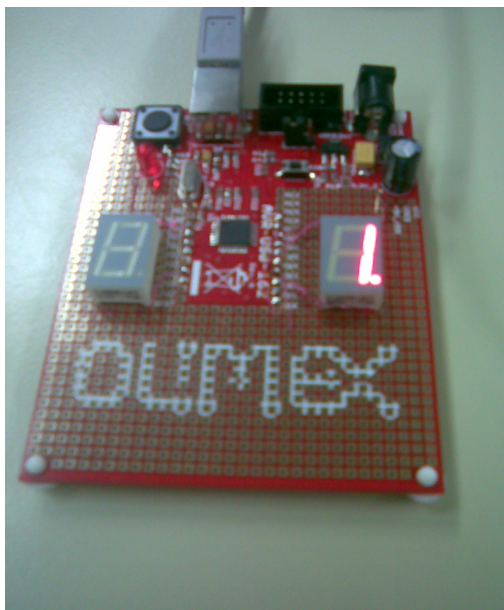


Bild1

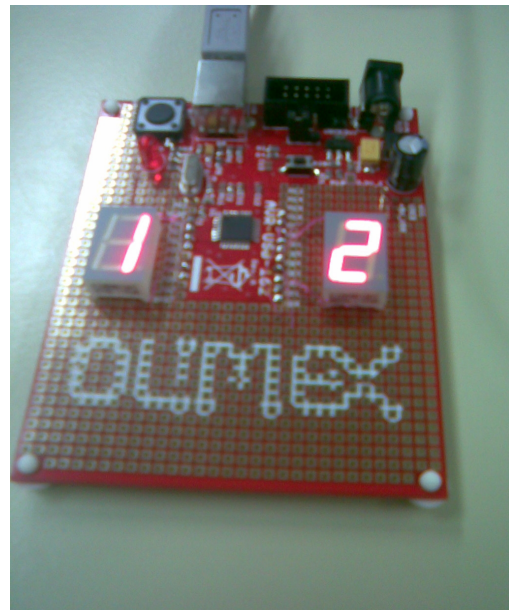


Bild2

Hier ist z.B die erste gezogenen Zahl die 12.

Anschließend kommt die „2.“, dann die 2. gezogene Zahl. Dies geht weiter bis zur 6. Zahl. Danach beginnt die Anzeige wieder bei der ersten Lottozahl. Die Anzeigen bleiben jeweils 2 Sekunden am Display stehen.

Durch Drücken des Restbutton kann das Spiel neu gestartet werden.

## Der Quellcode:

*/\*Copyright (c) 2010 Hubert Schlenk*

*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 Hubert Schlenk*

*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.\*/*

```
#include <avr/io.h>
```

```
#define F_CPU 1000000
```

```
#include <util/delay.h>
```

```
#define ANZAHL_STABIL 1000
```

```
void entprellen(void);
```

```
void check_button (void);
```

```
void anzeigen (void);
```

```
// Rechte Siebensegmentanzeige 0 1 2 3 4 5 6 7 8 9  
static unsigned char ziffer1[] = {0x77,0x14,0xB3,0xB6,0xD4,0xE6,0xE7,0x34,0xF7,0xF6};
```

```
// Linke Siebensegmentanzeige 0 1 2 3 4  
static unsigned char ziffer2[10][2]={{0x4E,0xC0},{0x48,0x00},{0x0D,0xC0},{0x4D,0x80},{0x4B,0x00}};
```

```
// Leeres Array für die gezogenen Zahlen
```

```
static unsigned char erg [6][3] = {};
```

```
int main (void)
```

```
{
```

```
int count=0, i=0,j=0;
```

```
DDRB = 0xff;
```

```
DDRC = 0xC0;;
```

```
DDRD=0x6F;
```

```
PORTD=0x80;
PORTD &= ~(1<<7);
```

```
while(1)
{
```

```
    while (count <6)
    {
```

```
        for (j=0; j<5; j++)
```

```
        {
            PORTC = ziffer2[j][1];
            PORTD = ziffer2[j][0];
```

```
            for(i=0; i<10; i++)
            {
```

```
                PORTB = ziffer1[i];
```

```
                _delay_ms(01);
```

```
                if(!(PIND &(1<<PIND7)))
                {
```

```
                    check_button();
```

```
                    if (i == 0 && j == 0)
                    {
```

```
                    }
```

```
                    else
                    {
```

```
                        erg [count][0] = ziffer1[i];
                        erg [count][1] = ziffer2[j][1];
                        erg [count][2] = ziffer2[j][0];
```

```
                        count ++;
```

```
                    }
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

```
    anzeigen ();
```

```
}
```

```
return 0;
```

```
}
```

```
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;
```

```
        }
```

```
    }
```

```
}
```

```

void check_button()
{
    if(PIND&(1<<7))
    {
        return;
    }
    entprellen();

    while((PIND&(1<<7))==0);
    entprellen();
}
void anzeigen (void)
{
    int count, count1;
    for (count = 0, count1 = 1; count <6; count ++, count1++)
    {
        PORTB = ziffer1 [count1];
        PORTB |= (1<<PIN3);
        PORTD =!0x4F;
        PORTC =!0xc0;
        _delay_ms(2000);
        PORTB = erg [count][0];
        PORTC = erg [count][1];
        PORTD = erg [count][2];
        _delay_ms(2000);
    }
}

```