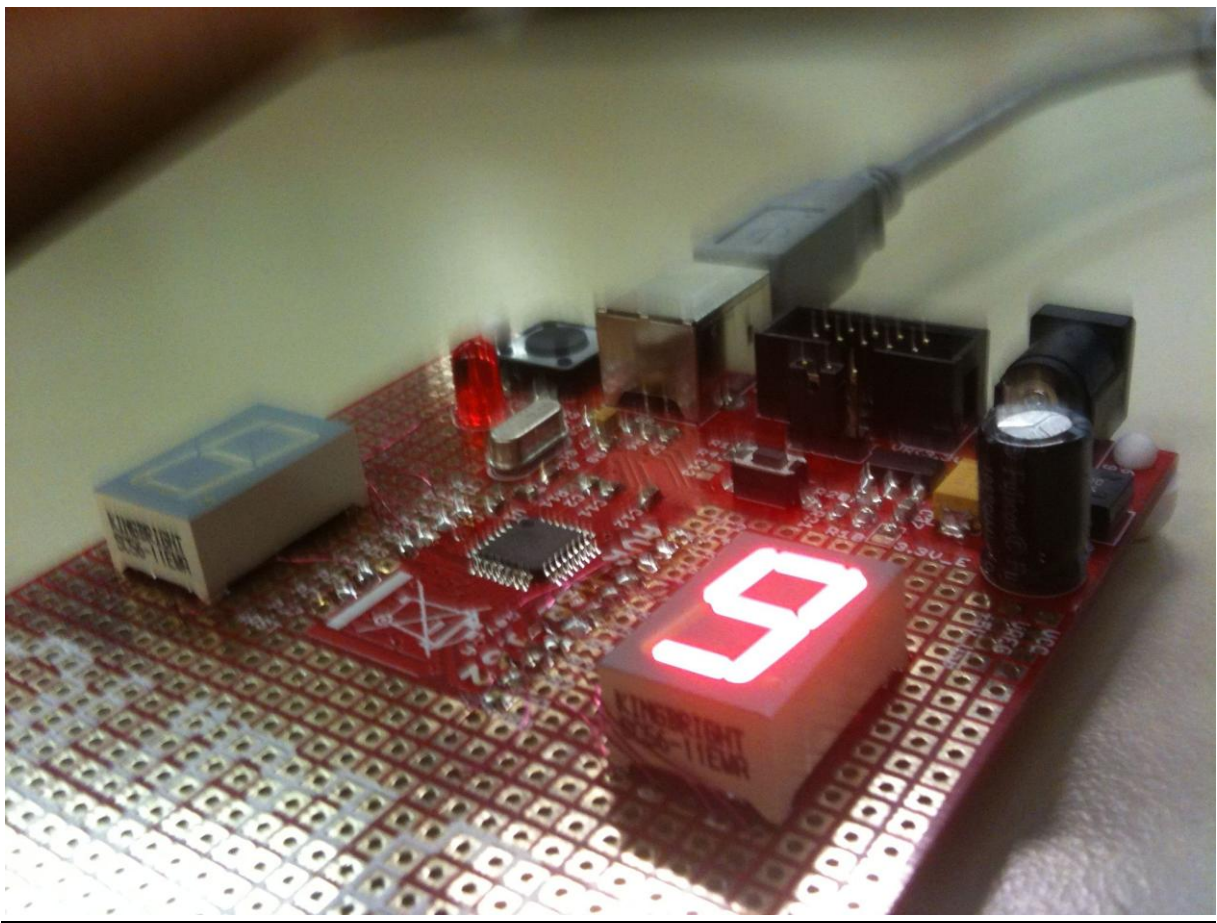


Projektarbeit: „Roulette Spiel „

auf dem Atmega162

Im Fach Datenverarbeitungstechnik



1. Aufgabestellung

Die Aufgabe verlangte es ein Reaktionsspiel auf einem Olimex AVR USB 162 zu entwerfen. Auf dem Entwicklerboard Olimex AVR USB 162 stehen zwei 7-Segment-Anzeigen, eine LED und ein Taster zur Verfügung.

2. Entwicklungsumgebung

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-Interface für Programmierung und USARTKommunikation integriert.

Die Entwicklerplatine AVR-USB-162 von Olimex übernimmt die Spannungsversorgung des Controllers, zusätzlich sind ein Reset-Button, 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 per Fädeldraht auf Port B, Port C und Port D angelötet, die genaue Belegung der Ports finden Sie unter Punkt 5.

Software:

- AVR-Studio 4
- Flip 3.4.1

3. „Roulette Spiel“

Als Spielidee diente uns das Prinzip des klassischen Roulette. Der Spieler wählt zuerst anhand der rechten 7-Segmentanzeige eine Zahl aus. Anschließend beginnt die linke Anzeige in hohem Tempo fortwährend von 0 bis 9 zu zählen. Nun muss durch geschicktes Drücken des Tasters eine Übereinstimmung der beiden Anzeigen erzielt werden. Ist dies Geglückt leuchten beide Ziffern zehn mal auf und das Spiel beginnt von neuem.

Quellcode:

*/*Copyright (c) 2010 Marco Ullrich, Tobias Stirnweiss*

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 Marco Ullrich, Tobias Stirnweiss

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./**

```

#define F_CPU 1000000
#include <avr/io.h>
#include <util/delay.h>

//Funktionen
void display2(int a, int b, int c,int d,int e,int f, int g);
void display (int);

int main(void)
{

    DDRB = 0xFF;
    DDRD = 0xFF;
    DDRC |= (1<<DDC6) | (1<<DDC7);

    char arr[10]={0x77,0x14,0xB3,0xB6,0xD4,0xE6,0xE7,0x34,0xF7,0xF6};
    int wert1,wert2, count=0;

    for(;;)
    {
    //Output
    PORTD = 0x00;
    PORTC = 0x00;
    PORTB = 0x00;

    //Input
    DDRD &=~(1<<DDD7); //Pin 7 = Taster
    PORTD |=(PORTD7); //Pull Up

    //rechte Anzeige hochzählen
    for(;;){
        PORTB = arr[count%10];
        count++;
        _delay_ms(500);

        //Ziel Wert speichern
        if(!(PIND&(1<<PORTD7))){
            _delay_ms(250);
            wert1 = (count%10);
            count=0;

            //Linke Anzeige hochzählen
            for(;;){

                display(count%10);
                count++;
                _delay_ms(100);

                //Getroffenen Wert überprüfen
                if(!(PIND&(1<<PORTD7))){
                    _delay_ms(250);
                    wert2 = (count%10);
                    count = 0;
                }
            }
        }
    }
}

```



```

return;
}

void display2(int a, int b, int c,int d,int e,int f, int g)
{
    int i = 0;
    int array[7]={a,b,c,d,e,f,g};

    //RESET
    PORTD = 0x00;
    PORTD |= (1<<PORTD7);
    PORTC = 0x00;

    //Setze Ports
    for(i=0;i<7;i++){
        if(array[i]==1){
            switch(i){
                case 0:
                    PORTD |= (1<<PORTD2);
                    break;
                case 1:
                    PORTD |= (1<<PORTD3);
                    break;
                case 2:
                    PORTD |= (1<<PORTD6);
                    break;
                case 3:
                    PORTC |= (1<<PORTC7);
                    break;
                case 4:
                    PORTC |= (1<<PORTC6);
                    break;
                case 5:
                    PORTD |= (1<<PORTD1);
                    break;
                case 6:
                    PORTD |= (1<<PORTD0);
                    break;
            }
        }
    }
    return;
}

```