Fachbereich Ingenieurwissenschaften I
Studiengang Informations- und Kommunikationstechnik (M)

Fach: Leiterplattendesign und Technologie
Bericht: Elektronischer Würfel

Teilnehmer: Brahima Drabo (538303)
Prabesh Nepal (542314)
Linda Kenmogne (547034)

Dozent: Prof. Uwe Metzler

Inhaltsverzeichnis

Abbildungverzeichnis

1 Einleitung

Heutzutage spielt die Elektronik eine wesentliche Rolle in der Technik. Die Welt wird immer mehr digitale, z. B. in der Medizin, in der Industrie, sogar in der Landwirtschaft wird sehr öfter die Technologie der Leiterplatten verwenden. Im Rahmen unseres Studiums ist eine bestimmte Module angeboten: Leiterplattendesign und Technologie, damit wir selber Projekt anhand eines Lastenheftes umsetzen. Das Projekt, mit dem wir uns beschäftigt haben, ist ein digitaler Würfel. Mit diesem Projekt haben wir uns im Bereich Embedded Systems weiterzubilden.

1.1 Zielsetzung

Ziel dieses Labors ist eine Platine zu entwickeln auf Anforderungen des Auftraggebers. Es wird ein elektronischer Würfel mit Leuchtdioden(LEDs) hergestellt Die Anzeige der LED soll genau wie bei einem echten oder realen Würfel mit den entsprechenden Würfelaugen dargestellt werden. Der Würfel wird mit einem Taster bedienbar sein, er soll nach 15 Sekunde allein in Sleep Modus (Standby-Modus) ausgeschaltet werden und muss durch Drücken des Tasters sich automatisch eingeschaltet und den letzten Wurf angezeigt werden. Die Stromversorgung wird durch eine Lithium-Batterie erfolgen. Diese Batterie soll eine Standfestigkeit von 5000 Würfeln haben.
Um den Würfel zu entwickeln, wird einerseits die CAD-Software Proteus für der Schaltung- und Layout-Design und für die Simulation verwendet und andererseits wird für die Programmierung des PICs die Software MPLAB von Microchip benutzt.

1.2 Vorgehensweise

Zuerst wird ein Plan für die Entwicklung der Leiterplatten erstellt, dann werden wir mit der CAD-Software Proteus beschäftigen, um den Umgang zur Platinen Erstellung zu erleichtern und wird eine Preisliste der Bauelemente entsprechend der Anforderungen angefertigt.

Danach wird die Schaltung in ARES UND DAS Platinenlayout in ISIS (Software von Proteus) erstellt. Dabei ist zu beachten, dass die Layout-Maße der Bauteile den Maßen der realen Bauteile entsprechen. Sobald die Schaltung und das dazugehörige Layout vollständig erstellt wurden, wird in MPLAB X der Quellcode für den Würfel auf Basis des Lastenheftes programmiert. Dieser wird dann in eine .hex-Datei konvertiert und in Proteus eingebunden, um die Entwicklung zu testen.

1.3 Zielgruppe

Die Anzusprechenden Zielgruppen sind Personen des Heimbereichs, sowohl Kinder als auch Erwachsene mit und ohne technisches Interesse in verschiedenen Altersgruppen, aber auch Personen im Rentenalter sollen als Zielgruppe sich angesprochen fühlen.

2 Produktfunktionen

a) Anzeige: /**LF01**/ Die Würfelaugen sollen wie bei einem richtigen Würfel durch Leuchtdioden dargestellt werden.

/**LF02**/ Während des Würfelvorgangs soll die Anzeige animiert werden.

b) Bedienung: /**LF10**/ Der Würfel wird durch einen Taster gestartet

c) Stromversorgung: /**LF20**/ erfolgt durch eine Lithium Batterie

/**LF22**/ nach 15 Sekunden schaltet sich der Würfel automatisch aus.

/**LF25**/ durch Drücken des Tasters schaltet sich der Würfel automatisch an und zeigt den letzten Wurf an.

d) /**LF30**/ Die Programmierung des Controllers soll mit dem Link Connector erfolgen.

3 Produktdaten

a) Anzeige: /**LD10**/ SMD LEDs

b) Bedienung: /**LD20**/ Taster auf der Platine

c) Stromversorgung: /**LD30**/ Lithium Batterie

d) Programmierung: /**LD40**/ Link Connector

4 Produktleistungen

Batterie: /**LL01**/ Die Batterie soll bis zu 5.000 Würfelvorgänge halten.

5 Qualitätsanforderungen

Sicherheit elektrischer Geräte für den Hausgebrauch und ähnliche Zwecke: VDE 0700 Teil 26, DIN EN 60335-2-26.

6 Produktumgebung

Für unseres Projekt sind die SMD-Bauelemente erforderlich. Diese Bauelementen sind auf der Oberfläche der Leiterplatten montiert, verglichen mit durchkontaktierten Bauelementen (THT), die durch Bohrungen auf der Platine gesteckt sind.

6.1 Hardware

Wir haben der PIC12F683 ausgewählt, das ist ein Mikrocontroller Microchip-Familie. Wir werden hier sieben LEDs verwenden, mit denen wir die Augen oder die Werte des Würfels anzeigen können. Da das Mikrocontroller-Datenblatt jedoch anzeigt, dass nur vier Pins zur Verfügung für die Zuweisung freie stehen, sollte ein neues LED-Display-Konzept in der Betrachtung gezogen werden.

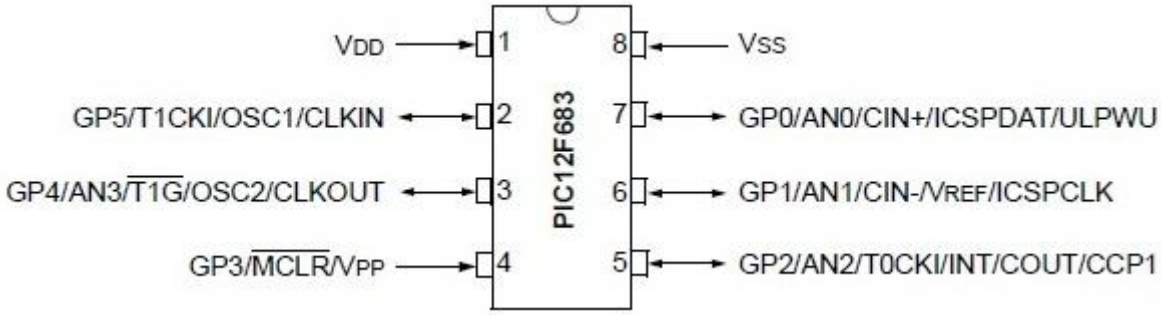

Abbildung 1: SMD PIC12F683

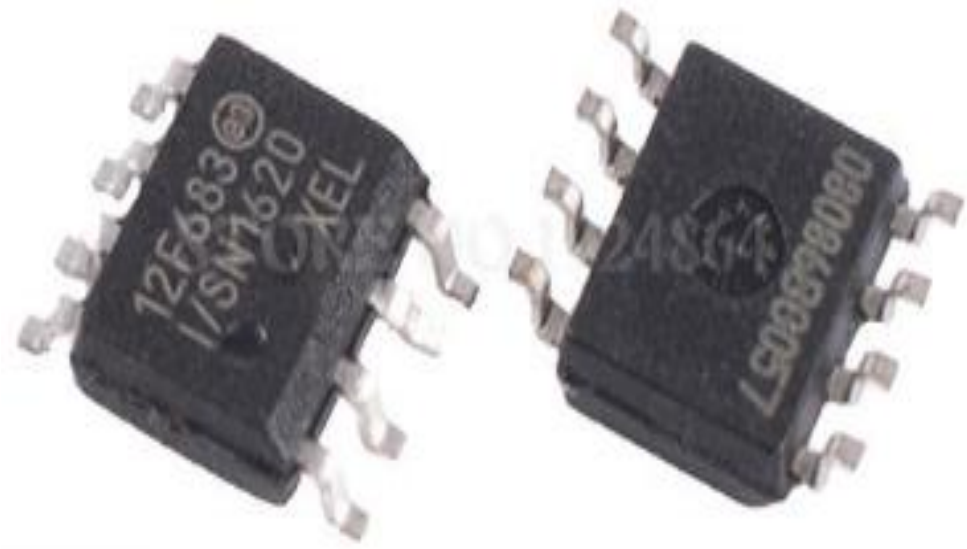

Abbildung 2: SMD PIC12F683

6.1.1 Liste der Bauteile

Bauteile	**Details**	**Stückzahl**	**Preis (einzeln) €**	**Preis (gesamt) €**
Microcontroller	PIC12F683	1	1,20	1,20
Rote LED	SMD LED OSRAM LS L29K	7	0,19	1,33
Taster	Drucktaster Würth Elektronik 431181015816	1	0,54	0,54
Vorwiderstand 100Ω	Dickschicht-Widerstand Samsung Electro-Mechanics RC3216F1624CS	1	0,20	0,20
Vorwiderstand 100Ω	Dickschicht-Widerstand Samsung Electro-Mechanics RC3216F1624CS	3	0,20	0,60
Pull-Up Widerstand 10kΩ	Dickschicht-Widerstand Samsung Electro-Mechanics RC3216F1624CS	1	0,20	0,20
Link Connector	PicKit 3 Tag Connect	1	0.20	0.20
Batterie 200mAh	Knopfzelle Panasonic CR 2032 Lithium HyCell	2	0,98	1,99
			Gesamt:	**6,23**

6.1.2 Berechnung der Widerstände

Für die Berechnung der Vorwiderstände wird die Betriebsspannung des Microcontrollers ( ) und die Betriebsspannung sowie der Betriebsstrom der LED's (+) benötigt. Mithilfe der Angaben und der Formel lassen sich dann die Widerstände berechnen.

$$U = R \cdot I$$

$$R = \frac{U_{ges} - U_{led}}{I_{led}}$$

Der PIC arbeitet mit einer Spannung von 3 V, die LED's mit jeweils 1,8 V. Des Weiteren geht aus dem Datenblatt der LED hervor, dass sie eine optimale Leistung bei 2 mA erbringen. Nun müssen insgesamt 2 Vorwiderstände ermittelt werden. Wie in Abschnitt 2.1 beschrieben wird ein Widerstand für eine LED konzipiert, die anderen für zwei parallel geschaltene LED's. Bei der Parallelschaltung der LED's bleibt die Spannung gleich. Lediglich die Stromstärken der LED's werden miteinander addiert. Daraus ergeben sich dann die folgenden Werte.

Vorwiderstand für eine LED:

$$R1 = \frac{U_{ges} - U_{led}}{I_{led}}$$
$$= \frac{3V - 1,8V}{16\,mA} = 75\Omega$$

Vorwiderstand für 2-Reihen geschaltete LEDs:

$$Rr = \frac{U_{ges} - 2(U_{led})}{I_{led}} = \frac{3V - 1,8V}{16mA} = 50\Omega$$

Ein Pullup- Widerstand wird dazu verwendet, einen Eingang auf einen definierten Wert zu "ziehen". Normalerweise befindet sich der Eingang im Zustand "schwebend/hochohmig", welcher sich irgendwo zwischen High und Low befindet. Es kann nun passieren, dass kurzzeitig mal ein Wert über- oder unterschritten wird und der Eingang plötzlich ein High- oder Low Signal bekommt. Dies führt dann zu unerklärlichen und unregelmäßig auftretenden Fehlern. Deshalb wird nun zwischen VCC und dem Eingang ein Widerstand eingesetzt, der Pullup-Widerstand heißt. Bei geschlossenem Taster wird nun der Strom über den Pullup-Widerstand nach GND fließen und der Input liegt auf GND (0V).

Um die programmierte Software auf dem Mikrocontroller zu spielen, benötigen wir einen Verbindungsstecker auf der Schaltung der sogenannte Link Connector. Die Stromversorgung des Würfels wird durch eine Batterie mit der zugehörigen Knopfzelle bereitgestellt.

6.1.3 Berechnung der Anzahl der Würfelvorgänge

Der Wert der verbrauchten Energie beträgt 50nA pro Stunde, wenn sich der Mikrocontroller im Standby-Modus befindet.

Würfel Vorgang ca. 2 Sekunden
Ausschalt der Würfel nach ca. 15 Sekunden
4 Würfelvorgänge in sechzig Sekunde
280 Würfelvorgänge in einer Stunde.
Somit verbraucht der Würfel für 280 Würfelvorgänge ca. 16,03mA.
$Würfelvorgänge$= 216−($Frequenz\ MikrocontrollerVorteiler * Frequenz$)−1=6800 $Vorgänge$

6.1.4 LED-Kombinationen für den Augenansicht

Beim Drücken des Taster wird jedes mal eine Zufällige Zahl gewürfelt, deswegen können wir nicht bestimmen welche Zahl wird das nächste sein. Es kann auch sein das selbe Zahl mehrmal vorkommt.

Fall 1 : LED 4 leuchtet (GP0 = D4)
Fall 2 : LED 3 + LED 6 leuchten (GP2 = D3 + D6)
Fall 3 : LED 4 + LED 3 + LED 6 leuchten (GP0+GP2)
Fall 4 : LED 2 +LED 5 + LED 3 + LED 6 leuchten (GP1+GP2)
Fall 5 : LED 4 +LED 2 + LED 5 + LED 3 + LED 6 leuchten (GP0+GP1+GP2)
Fall 6 : LED 2 + LED 5 + LED 3 + LED 6 + LED 1 + LED 7(GP1+GP2+GP5)

Fall 1 : LED 4 leuchtet (GP0 = D4)

Fall 2 : LED 3 + LED 6 leuchten (GP2 = D3 + D6)

Fall 3 : LED 4 + LED 3 + LED 6 leuchten (GP0+GP2)

Fall 4 : LED 2 +LED 5 + LED 3 + LED 6 leuchten (GP1+GP2)

Fall 5 : LED 4 +LED 2 + LED 5 + LED 3 + LED 6 leuchten (GP0+GP1+GP2)

Fall 6 : LED 2 + LED 5 + LED 3 + LED 6 + LED 1 + LED 7(GP1+GP2+GP5)

6.2 Schaltplan

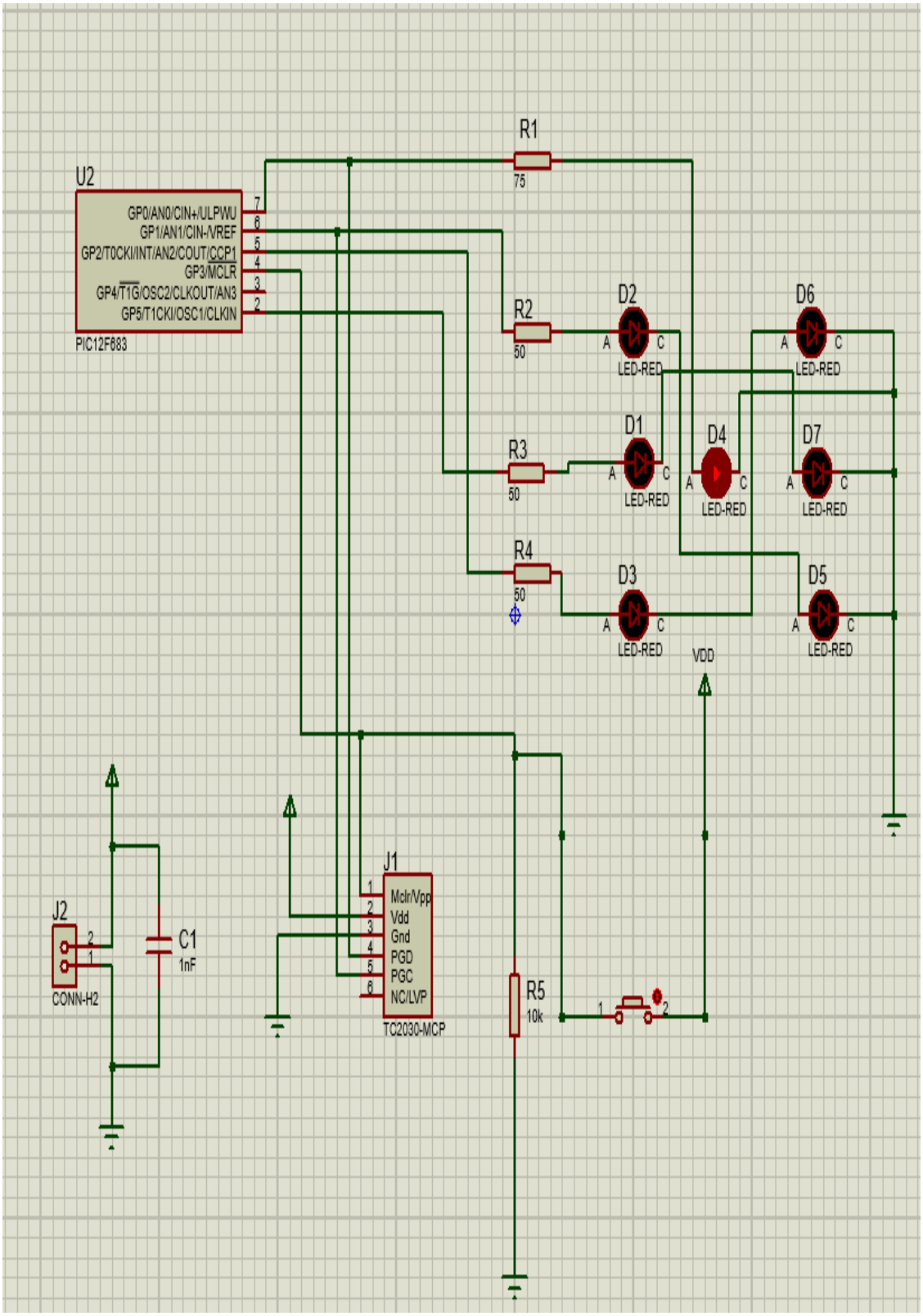

Abbildung 3: Schaltplan

6.3 PCB-Layout

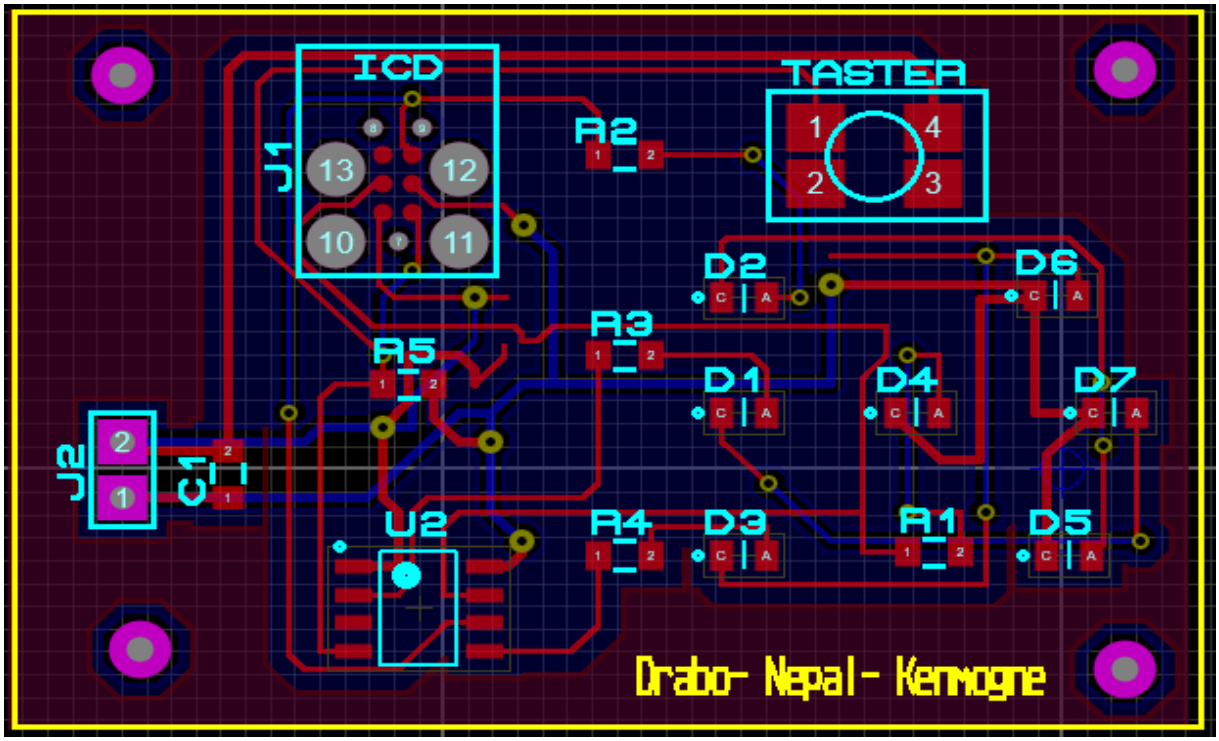

Abbildung 4: PCB Layout

6.4 3D-Ansicht

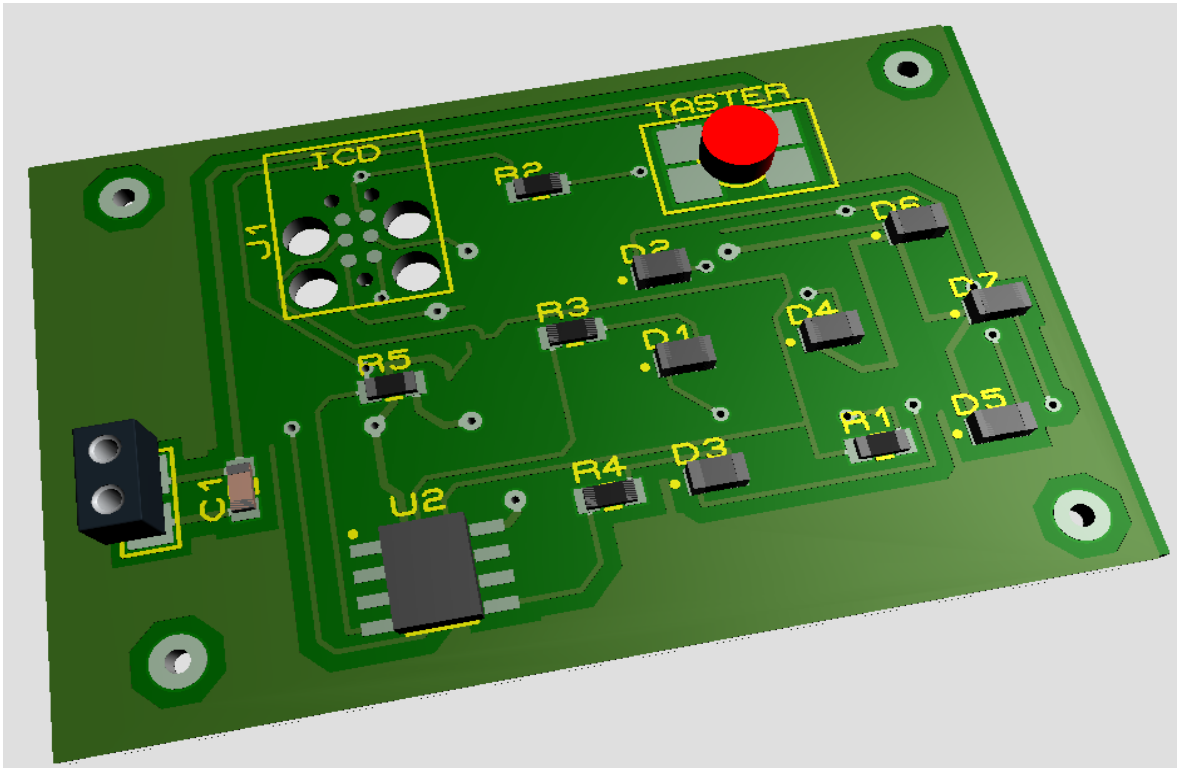

Abbildung 5 : 3D Ansicht der Platine

6.5 Programmablauf

Die Platine läuft mit drücken des Tasters. Unten in der Abbildung haben wir gezeigt wie der Platine reagiert nach dem drücken des Tasters.

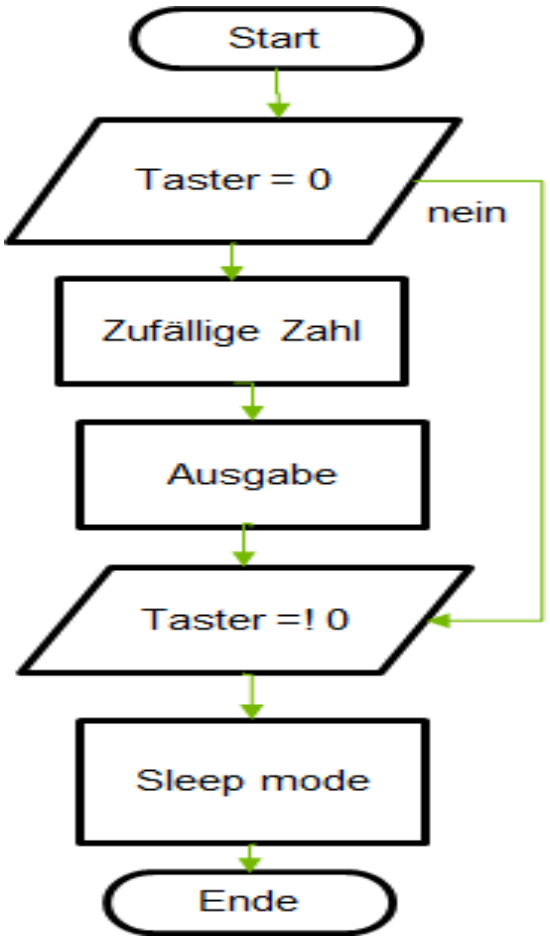

Abbildung 6: Flowchart der Programmablauf

6.6 Software

Das Programm wurde in C-Sprache geschrieben. Dafür haben wir die Entwicklungsumgebung MPLABX benutzt. Nach der Kompilierung des programmierten Quellcodes wird automatisch eine .hex-Datei generiert, die anschließend in Proteus eingebunden wird, um die Schaltung mit dem Programm zu testen.

6.7 Schnitstellen

Für den Würfel ist eine ext. Programmierschnittstelle vorgesehen, so dass man bei einem Fehlverhalten das Problem schnell erkennen kann. Über diese Schnittstelle kann der Würfel an einen Computer angeschlossen werden und auf etwaiges Fehlverhalten hin getestet werden.

7 Benutzerschnittstelle

Das Produkt funktioniert beim Drucken des einzigen Tasters.

Benutzer Taster Animierte LEDs

Ablauf bei der Taster Betätigung

- Tasterdruck
- Darstellung des Letzen Wurfs
- Animation des Würfels Vorganges
- Anzeige der gewürfelten Augen
- Auschaltvorgang in 15. Sekunden ohne Tasterdruck

8 Qualitätszielbestimmung

Der Würfel soll nach der VDE 0700 Teil 26, DIN EN 60355-2-26 für elektronische Geräte im Hausgebrauch und ähnliche Zwecke entwickelt werden. Außerdem soll der Würfel Funktionen Benutzerfreundlich sein, da die Altersgruppen von 6-99 Jahre angesprochen werden soll. Zusätzlich soll der Würfel eine Zuverlässigkeit für den Würfelvorgang besitzen, d.h. dass der Würfel nicht permanent dasselbe Zahl würfelt. Die SMD-LED soll so ausgewählt werden, dass sie mindesten 5000 Würfel Vorgänge ohne Austausch von Batterie erfolgen. Nach der Rechnung können wir feststellen das die Batterie etwa 6800 Wurf würfeln kann.

9 Anhang

C Datei

```
// PIC12F683 Configuration Bit Settings

// 'C' source line config statements

// CONFIG
#pragma config FOSC = INTOSCIO  // Oscillator Selection bits (INTOSCIO oscillator: I/O function on RA4/OSC2/CLKOUT pin, I/O function on RA5/OSC1/CLKIN)
#pragma config WDTE = OFF       // Watchdog Timer Enable bit (WDT disabled)
#pragma config PWRTE = OFF      // Power-up Timer Enable bit (PWRT disabled)
#pragma config MCLRE = OFF      // MCLR Pin Function Select bit (MCLR pin function is digital input, MCLR internally tied to VDD)
#pragma config CP = OFF         // Code Protection bit (Program memory code protection is disabled)
#pragma config CPD = OFF        // Data Code Protection bit (Data memory code protection is disabled)
#pragma config BOREN = OFF      // Brown Out Detect (BOR disabled)
#pragma config IESO = OFF       // Internal External Switchover bit (Internal External Switchover mode is enabled)
#pragma config FCMEN = OFF      // Fail-Safe Clock Monitor Enabled bit (Fail-Safe Clock Monitor is disabled)

// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.

#include <stdio.h>
#include <stdlib.h>
#include <xc.h>
#include <math.h>
#include <time.h>

#define _XTAL_FREQ 4000000
#define taster GPIObits.GP3

/*
*
*/


//int zufall = 0;
int augenzahl = 0;
int count = 0;
```

```
void timer1(void)
{
   T1CONbits.TMR1ON = 1;   //Enables Timer1
   T1CONbits.TMR1CS = 0;  //Internal clock (FOSC/4)
   T1CONbits.T1CKPS1 = 1;  // Timer1 Input Clock Prescale (1:8)
   T1CONbits.T1CKPS0 = 1;

   TMR1L = 0;
   TMR1H = 0;   //Register

   PIE1bits.TMR1IE = 1;    //Enables the Timer1 overflow interrupt
   INTCONbits.GIE = 1;       //Global Interrupt Enable bit
   INTCONbits.PEIE = 1;     //Peripheral Interrupt Enable bit
   INTCONbits.GPIE = 1;     //Enables the GPIO change interrupt


}

void __interrupt() myISR(void)
{

    if(PIR1bits.TMR1IF == 1)    // (INTF) INTF Timer1 Overflow Interrupt Flag bit
    {

       count = count + 1;
       TMR1L = 0;                        //Reset TMR1-Register
       TMR1H = 0;


       PIR1bits.TMR1IF = 0;  //Timer1 has not overflowed (reset timer1 flag)

    }

}


void Wahl()
{

    GPIO = 0b11001000;        // Leds ausschalten
    augenzahl = (rand()%6)+1;  // Zufallszahl
    __delay_ms(5);
    switch (augenzahl)        //Ausgabe der Zahl
    {
     case 1:
         GPIO = 1;
         //__delay_ms(500);
```

```
            break;
        case 2:
            GPIO = 4;
            //__delay_ms(500);
            break;
        case 3:
            GPIO = 5;
            //__delay_ms(500);
            break;
        case 4:
            GPIO = 6;
            //__delay_ms(500);
            break;
        case 5:
            GPIO = 7;
            //__delay_ms(500);
            break;
        case 6:
            GPIO = 38;
            //__delay_ms(500);
            break;
        }
     __delay_ms(5);
}

int main(void) {

    TRISIO = 0b11001000;    //Gp0,Gp1,Gp2,Gp4,Gp5 als Ausgang definiert


    ANSEL = 0;          //Disable
    CMCON0 = 7; // Disable ADC comparator
    ADCON0 = 0;
    IOC = 0;
    CMCON1 = 0;
    timer1(); // Timer Aufruf
    while(1){


         if(INTCONbits.GPIE == 0) //Disables the GPIO change interrupt
         {

           SLEEP();
           GPIO = augenzahl;
           //LedLeuchten();
           INTCONbits.GPIE = 1 ; //Enables the GPIO change interrupt
           __delay_ms(100);
         }
```

```
      if(taster == 0)                    //Wenn Taster betätigt wird
        {
          count=0;                    // Zähler zurücksetzten
          Wahl();
          //LedFolge();                    //Aufruf Funktion: Wahl
        }
        else
          {
           if (count >= 24)              //(24)Wenn 15 Sekunden vorbei sind
              {
                count = 0;                 // Zähler zurücksetzten
                GPIO = 0b00000000;              //Led's ausschalten
              }

     }

   }
}
```

Quellen

1. Datensheet der PIC12F683
2. Skript – Prof. Skilandat
3. Pflichtenheft