VIDEO ONLINE

40x 10-mm-LEDs

Quellcode frei per Download

16 programmierte Lichtmuster

Programmierbares Grafik-LC-Display

Mitteilsamer Hingucker – LED-Blinkherz LBH40

Ob als Dekoobjekt mit Begrüßungstext für Gäste an der gläsernen Haustür, als „sprechendes" Geschenk für die Liebste oder kleine elektronische Pinnwand – Einsatzbereiche für dieses Elektronik-Dekoprojekt lassen sich viele finden. Das LBH40 erzeugt mit 40 in Herzform angeordneten LEDs aus 16 wählbaren Mustern abwechslungsreiche Lichteffekte. Zusätzlich sind mit einem LC-Display 10 frei programmierbare Lauftexte anzeigbar. Über eine Programmierschnittstelle kann das Objekt auch als Lern- und Experimentiersystem für die AVR-Programmierung eingesetzt werden.

Mehr als nur Deko

Elektronische Dekoobjekte stehen derzeit hoch im Kurs, etwa solche wie unser LED-Cube oder die LED-Matrix-Anzeigen. Dieses Projekt wird persönlich – es eignet sich nämlich hervorragend als Geschenk vom lötenden und programmierenden Elektroniker mit persönlicher Widmung. Die 40 in Herzform angeordneten LEDs können dabei mit 16 programmierten Lichtmustern betrieben werden. Zusätzlich sitzt zentral im Herz ein beleuchtetes Grafik-LC-Display, das frei nach Wunsch programmierbar ist. So sind 10 Lauftexte mit je bis zu 99 Zeichen speicherbar.

Die Bedienung und Programmierung der Texte erfolgt über 5 auf der Platine befindliche Tasten.

Wenn man die Platine des LBH40 genau betrachtet, fällt ein 10-poliger Wannenstecker ins Auge – eine ISP-Schnittstelle. Die macht das gesamte System zum Experimentier- und Lernsystem für die AVR-Programmierung.

Sämtliche Hardware dafür ist an Bord: LEDs, Tasten, ein Display und natürlich die erforderliche Stromversorgung. Und die Firmware des LBH40 ist offen gelegt und steht als komplette Datei in Form eines Atmel-Studio-Projekts zum Download bereit. Damit kann auch der Programmier-Anfänger unmittelbar in das Projekt einsteigen, sich zunächst über eigene Programm-Modifikationen einarbeiten, wesentliche Algorithmen anhand eines „lebenden" Beispiels kennen und verstehen lernen und schließlich eigene Programme schreiben. Auf diese Möglichkeit gehen wir am Schluss des Artikels detailliert ein.

Daten

Kurzbezeichnung:	LBH40
Versorgungsspannung:	7,5–16 VDC
Stromaufnahme:	500 mA max.
Umgebungstemperatur:	5–35 °C
Einsatzgebiet:	Wohngebäude/Wohnflächen wie Häuser, Wohnungen, Zimmer usw.
Abmessungen (B x H x T):	250 x 230 x 20 mm
Gewicht:	500 g

Bedienung

Das Blinkherz besitzt 5 Tasten zur Bedienung, wobei diese wie ein Steuerkreuz mit in der Mitte liegender Bestätigungstaste („OK“) aufgebaut sind.

Über die Bestätigungstaste lässt sich ein Menü aufrufen, in welchem mit den Pfeil-Tasten ▲ und ▼ navigiert werden kann. Der aktivierte Menüpunkt wird dabei invertiert dargestellt (Text hell, Hintergrund dunkel). Mit der Pfeil-Taste ▶ oder der in der Mitte liegenden Bestätigungstaste lassen sich die Menüpunkte auswählen. Mit der Pfeiltaste ◀ kehrt man zur nächst höheren Ebene zurück oder verlässt das Menü. In Bild 1 ist die komplette Menüstruktur zu sehen.

Auswahl von Blinksequenzen/Texten

Es werden Zahlen von 1 bis 16 (Blinksequenzen)/ 1 bis 10 (Texte) auf dem Display angezeigt, dabei sind aktivierte Texte/Sequenzen invertiert dargestellt (Zahl hell, Hintergrund dunkel).

Die zur Zeit änderbare Zahl wird mit einem kleinen Pfeil markiert. Mittels der Rechts-/Links-Tasten lässt sich der Pfeil auf eine andere Zahl bewegen. Mittels der Hoch-/Runter-Tasten lässt sich die Auswahl ändern (invertiert ausgewählt, nicht invertiert abgewählt). Mit der „OK“-Taste wird diese Auswahl gespeichert und in das Grundmenü zurückgekehrt. Änderungen werden erst nach Durchlauf der vorher aktiven Sequenz bzw. des vorher aktiven Textes sichtbar.

Texte ändern

Im Menü „Texte ändern“ muss zunächst der zu ändernde Text ausgewählt werden, dieses kann mit den Hoch-/Runter-Tasten ausgeführt werden. Die Nummer des zu ändernden Textes steht in der zweiten Zeile, darunter wird der Anfang des Textes als Vorschau angezeigt.

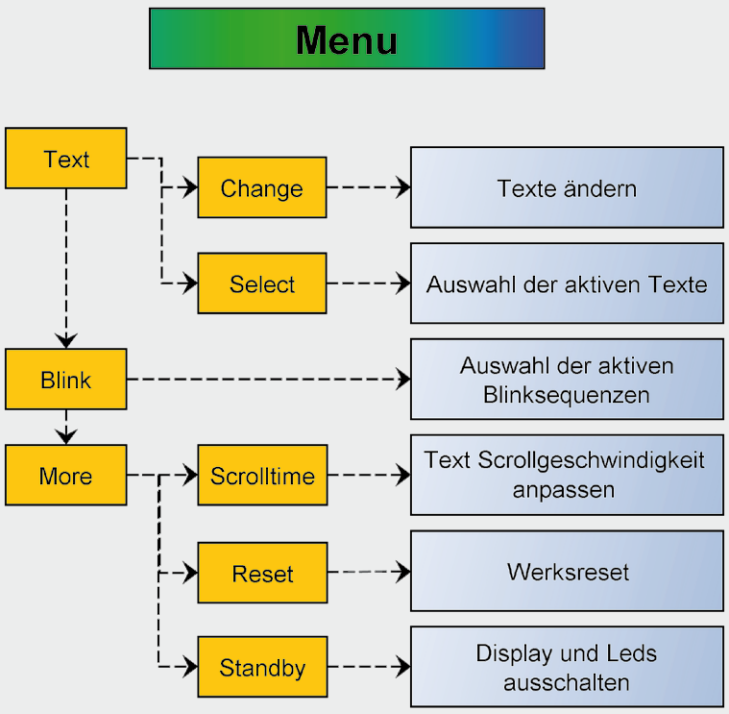


Bild 1: Der Menüaufbau zur Bedienung des LBH40

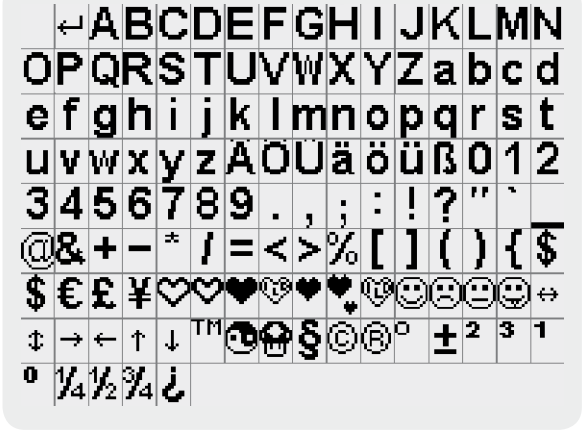


Bild 2: Der Zeichensatz für das Grafikdisplay

Nach Bestätigung mit der „OK“-Taste kann der Text geändert werden, dabei wird der aktuell änderbare Buchstabe invertiert dargestellt. Mittels der Links-/ Rechts-Tasten lässt sich ein anderer Buchstabe auswählen und über die Hoch-/Runter-Tasten ändert man den Buchstaben. Bild 2 zeigt dabei den zur Verfügung stehenden Zeichensatzvorrat.

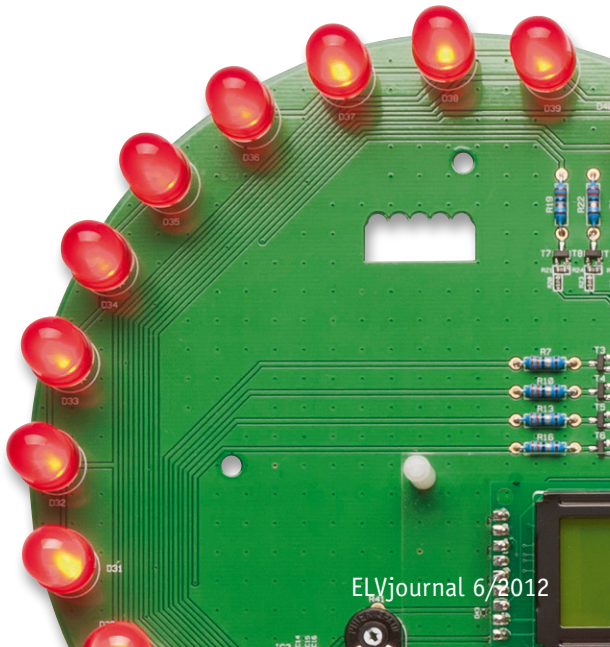
Wird die Hoch- bzw. Runter-Taste länger gedrückt gehalten, erfolgt so lange ein automatischer Durchlauf durch den Zeichensatz, bis die Taste wieder losgelassen wird. Texte können dabei bis zu 99 Zeichen enthalten. Das zweite Zeichen aus dem Zeichensatz (ENTER-Symbol) beendet einen Text, so dass alle nachfolgenden Zeichen gelöscht werden.

„More“

Unter dem Menüpunkt „More“ lässt sich die Scrollgeschwindigkeit des Textes ändern, dabei stehen Werte zwischen 5 und 20 zur Verfügung (die empfohlene Einstellung ist 7), wobei 20 die langsamste und 5 die höchste Geschwindigkeit darstellt.

Unter „Reset“ kann ein Werksreset durchgeführt werden, nach erneuter Bestätigung werden dabei alle gespeicherten Texte und Text-/Sequenzauswahlen wieder auf den Auslieferungszustand zurückgesetzt.

Mit „Standby“ lässt sich das LBH40 in einen Standby-Modus versetzen, wobei die LEDs und das Display ausgeschaltet werden. Nach Betätigung einer beliebigen Taste werden Display und LEDs wieder eingeschaltet.



Schaltung

Das Schaltbild des LBH40 (Bild 3) ist recht übersichtlich und besteht aus dem Prozessorteil mit den Bedienelementen, dem Display, der Spannungsversorgung und den LEDs samt Matrixansteuerung.

Zentrales Bauelement ist der Mikrocontroller IC1, welcher das gesamte Gerät steuert. Zur Speicherung der Texte und Einstellungen verfügt dieses IC über einen internen EEPROM-Speicher, der eine Speicherkapazität von 2 KByte zur Verfügung stellt. Die Firmware hingegen ist im Flash-Speicher des Controllers untergebracht.

Die Kondensatoren C4 bis C13 und C21 bis C29 dienen zur Spannungsstabilisierung, während R35 für einen definierten Pegel am Reset-Eingang und damit für einen definierten Start des Controllers sorgt.

Die 5 Tasten sind direkt an den Mikrocontroller angeschlossen, die jeweiligen I/Os sind intern mit Pullups beschaltet.

Über den Steckverbinder ST1, der die ISP-Programmierschnittstelle des Mikrocontrollers darstellt, können eigene Programme über einen ISP-Programmieradapter in den Controller programmiert werden.

Der Quellcode für das LBH40 wird am Schluss des Artikels teilweise erläutert und steht komplett auf der Produktseite (Webcode #1243) zum Download bereit.

Die Ansteuerung der 40 LEDs erfolgt in einer Matrix, welche aus 5 Gruppen (Spalten) zu je 8 LEDs (Zeilen) besteht. Die Zeilen werden am Controller-Port C angeschlossen, wobei die Transistoren T1 bis T8 als Treiber dienen. Der Strom für die LEDs wird über die jeweiligen Vorwiderstände R1, R4, R7, R10, R13, R16, R19, R22 definiert. Die Spalten werden (am Port D) ebenfalls mittels Transistoren angesteuert. Dabei werden die Spalten nacheinander aktiviert und entsprechend das Muster an den Zeilen angelegt. Dieses Verfahren wird auch Multiplexing genannt. Mehr dazu und zur Berechnung der LED-Vorwiderstände ist unter „Elektronikwissen“ nachzulesen.

Bei dem in diesem Projekt eingesetzten Display handelt es sich um ein Grafikdisplay, welches 122 x 32 Pixel besitzt. Dabei ist das Display in 2 Hälften unterteilt, die jeweils über einen eigenen Controller angesteuert werden. Diese beiden Controller werden über die Signale E1 und E2 jeweils während der Ansteuerung ausgewählt. Das Signal R/W dient zur Auswahl zwischen Lese- oder Schreibzugriff und A0 zur Unterscheidung zwischen Befehl oder Daten. Neben den Steuersignalen sind zusätzlich 8 Leitungen zur Datenübertragung vorhanden.

Das Display benötigt eine negative Kontrastspannung, die von dem Spannungsinverter IC3 und den Kondensatoren C14 bis C20 generiert

Elektronikwissen

LED-Multiplexing (Zeitmultiplex) und Widerstandsberechnung

Beim Multiplexing werden die LEDs in einer Matrix angeordnet, so können I/Os am Controller eingespart werden. Die LEDs sind in Spalten und Zeilen aufgeteilt, wobei jedoch immer nur die LEDs einer Spalte wirklich aktiv sind. Die Zeilen geben vor, welche LEDs in der Spalte aktiv sind, es werden die Spalten dann der Reihe nach durchgeschaltet und die entsprechenden Zeilen aktiviert. Geschieht dies mit einer ausreichenden Geschwindigkeit, wirkt es durch die Trägheit des menschlichen Auges wie ein vollständiges Bild (ab ca. 100 Hz flimmerfrei). Dabei handelt es sich um Zeitmultiplex (Time Division Multiplex), da die Informationen (LEDs) in zeitlicher Abfolge zusammengepackt werden.

Beim Multiplexing ist eine LED nicht dauerhaft eingeschaltet, sondern nur für kurze Intervalle. Um trotzdem die gleiche Helligkeit erreichen zu können, muss der Strom möglichst um den Faktor des Multiplexing (Faktor möglichst gering halten) gesteigert werden, dabei ist der maximal zulässige Pulsstrom der LED zu beachten.

Typische Angaben für LEDs:

- Pulsstrom meist 100 mA
- Duty-Cycle 1/10 bei 10 kHz

Dabei wäre eine LED dann für 10 µs eingeschaltet und anschließend 90 µs ausgeschaltet.

Im günstigsten Fall wird der Nennstrom der LED mit der Anzahl der Spalten multipliziert.

$$I_{Led_{Puls}} = I_{Led_{Nenn}} * N = 20mA * 5 = 100mA$$

Zu beachten sind dabei aber auch die Angaben zum Pulsstrom der LED im Datenblatt! Ebenso sollte beachtet werden, dass beim LBH40 die LEDs nicht mit 10 kHz, sondern 1 kHz geschaltet werden, weshalb der maximale Pulsstrom zu reduzieren ist.

Berechnung Vorwiderstand:

- N Anzahl der multiplexten Spalten
- U_B Versorgungsspannung
- U_{CEsat} Sättigungsspannung Transistor

$$R_V = \frac{U}{I} = \frac{(U_B - U_{CEsat} - U_{Led})}{(I_{Led_{Nenn}} * N)} = \frac{(5V - 0{,}7V - 1{,}7V)}{20mA * 5} = 26\,\Omega$$

Je nach Anzahl der eingeschalteten LEDs pro Spalte fällt über den Transistor aufgrund des höheren Stroms eine höhere Spannung an der CE-Strecke ab. Dieses hätte durch Verwendung von MOSFETs umgangen werden können, doch MOSFETs sind sehr empfindlich gegenüber Überspannungen am Gate. Da für das LBH40 kein Gehäuse vorgesehen ist, wäre somit dort ein erhöhter Aufwand zu betreiben, um die MOSFETs zu schützen. Um dieses zu vermeiden, wurden Transistoren eingesetzt, diese sind wesentlich robuster.

Der für die LED kritische Fall ist, dass am Transistor keine Spannung abfällt, deswegen sollte ein größerer Widerstand gewählt werden. Siehe zweite Berechnung mit U_{CEsat} = 0V :

$$R_V = \frac{U}{I} = \frac{(U_B - U_{CEsat} - U_{Led})}{(I_{Led_{Nenn}} * N)} = \frac{(5V - 0V - 1{,}7V)}{20mA * 5} = 33\,\Omega$$

Farbe	Durchlassspannung	Vorwiderstand
Rot	1,7 V	47 Ω
Gelb	2,1 V	47 Ω
Grün	2,2 V	47 Ω
Blau	4,0 V	15 Ω
Weiß	3,2 V	27 Ω

Je nach Helligkeit der LEDs sollte ein noch größerer Widerstand, als in der Tabelle aufgeführt, gewählt werden, da die LEDs sonst zu hell leuchten könnten und den Betrachter blenden würden.

Dem Bausatz sind für das als Zubehör angebotene LED-Set mit roten 10-mm-LEDs verschiedene Widerstandswerte beigelegt, um so in der Helligkeit variieren zu können, es empfiehlt sich ein Wert von 120 Ω.

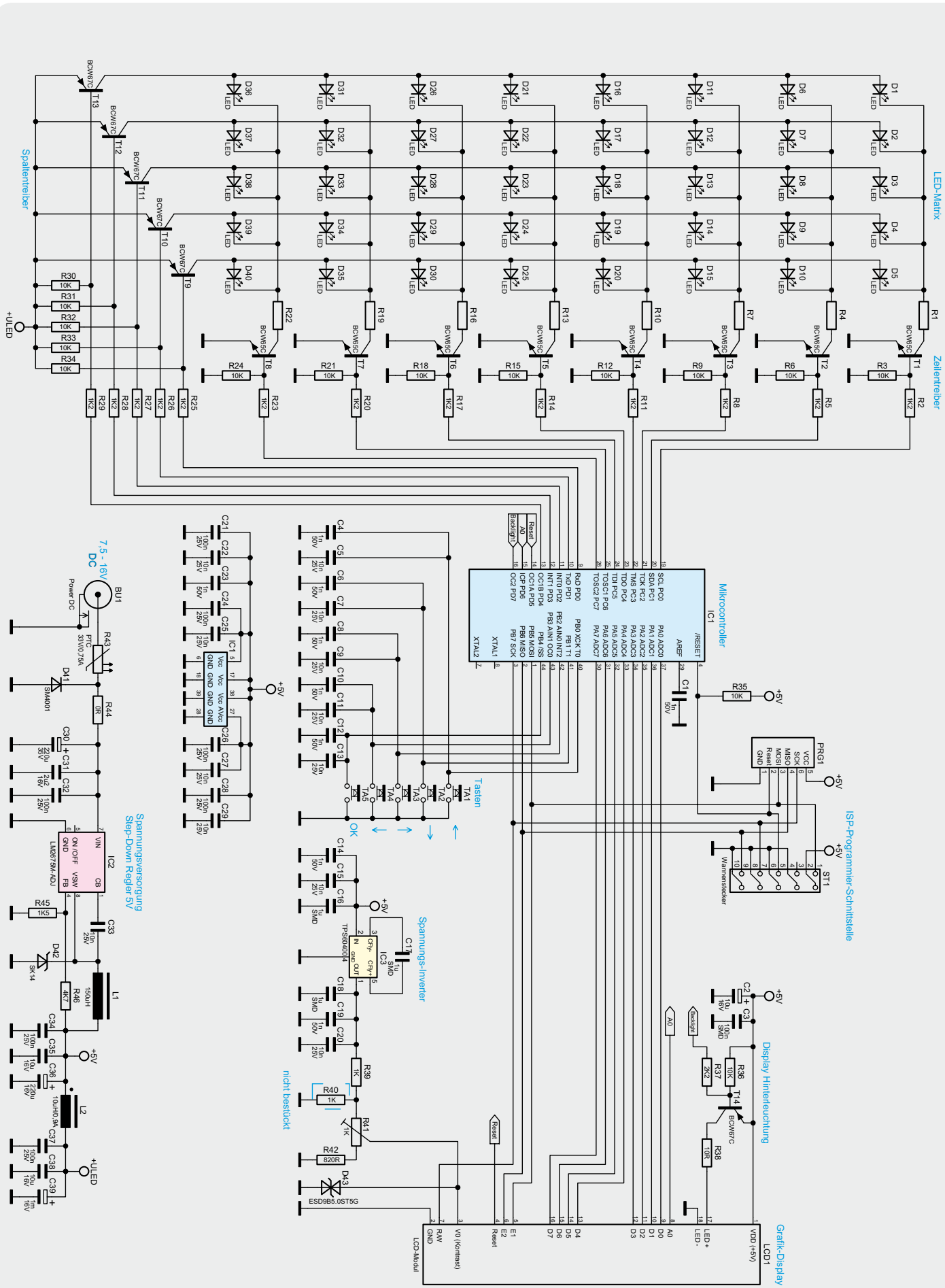


Bild 3: Die Gesamtschaltung des LBH40. Die Vorwiderstände der LEDs sind entsprechend der Beschreibung im Text zu dimensionieren.

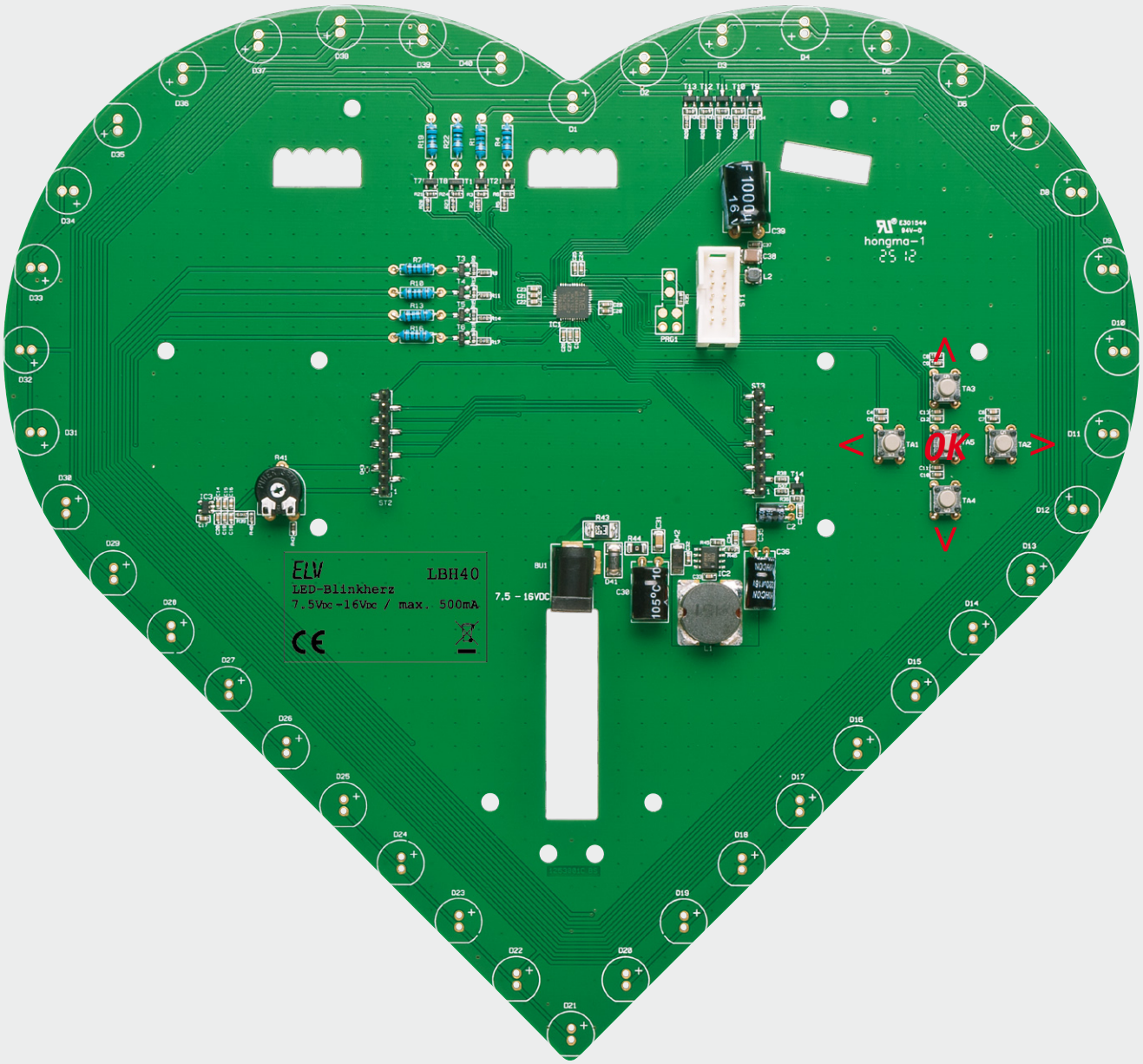


Bild 4: Die komplett (außer LC-Display und LEDs) bestückte Platine des LBH40 mit zugehörigem Bestückungsplan

wird. Der Spannungsteiler aus den Widerständen R39, R42 und dem Potentiometer R41 dient zur Einstellung des Kontrastes. Über den Transistor T14 lässt sich zusätzlich die Hintergrundbeleuchtung des Displays schalten, der Widerstand R38 begrenzt den Strom der Hintergrundbeleuchtung.

Die Versorgung der Schaltung übernimmt der Schaltregler IC2 zusammen mit den Komponenten L1, C33 und D42. Über den Spannungsteiler R45, R46 wird die Ausgangsspannung eingestellt, hier für 5 V. Als Sicherheitselemente sind R43 als Überstromschutz und D41 als Verpolungsschutz vorhanden. Die Kondensatoren C30 bis C32 und C34 bis C36 stabilisieren die jeweilige Spannung. Die Spule L2 und die Kondensatoren C37 bis C39 dienen zur Drosselung der Stromspitzen im LED-Schaltungsteil, dort treten je nach Blinkmuster hohe Laständerungen auf, welche den Schaltregler ohne diese Schutzmaßnahme übermäßig belasten würden.

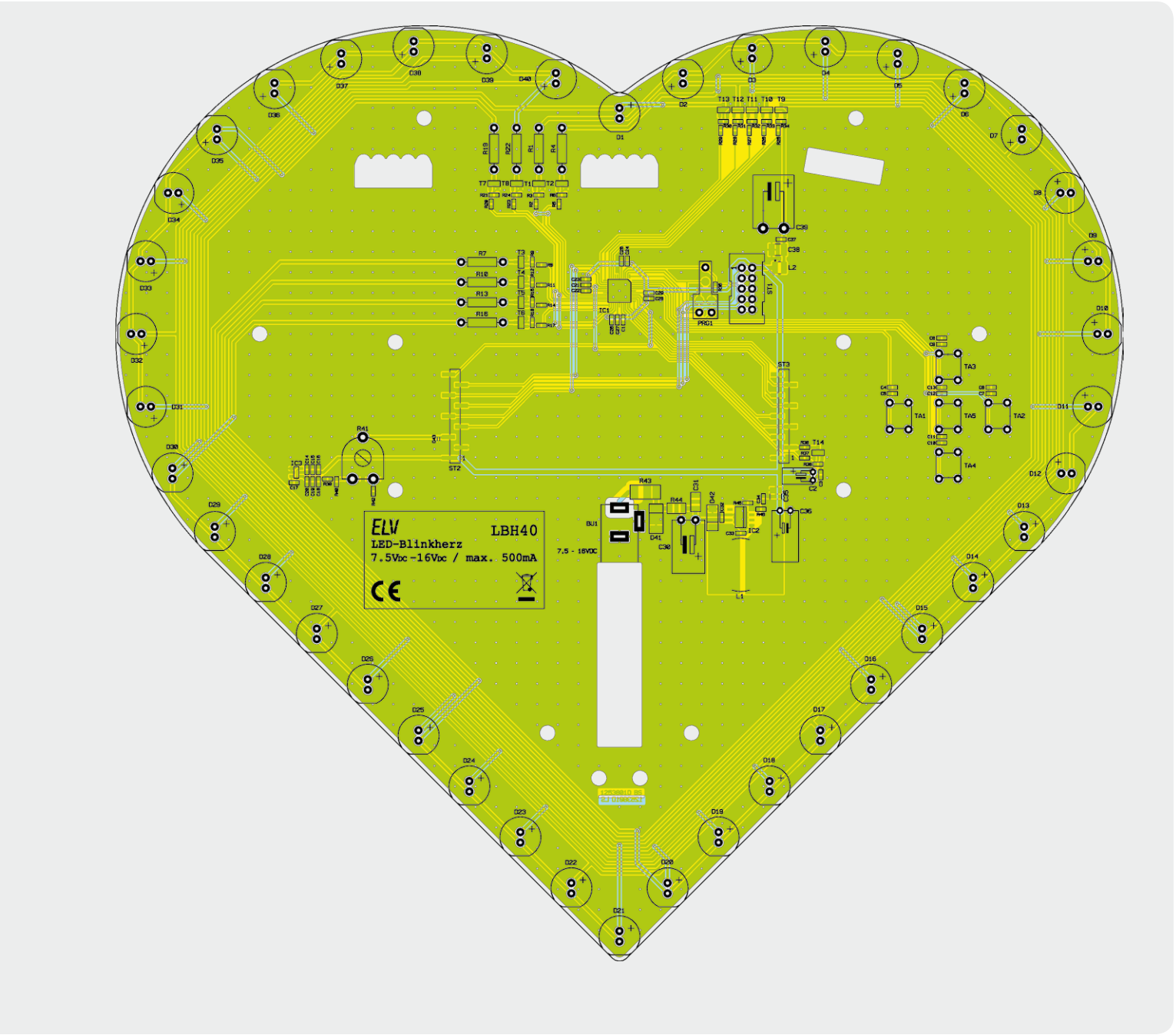
Nachbau

Der Aufbau des LBH40 erfolgt auf einer einseitig zu bestückenden, herzförmigen Platine, wobei die SMD-Bauteile bereits bestückt sind. Diese Bestückung ist lediglich auf Bestückungs- und Lötfehler zu kontrollieren.

So sind lediglich die bedrahteten Bauteile entsprechend Platinenfoto, Bestückungsplan (Bild 4), Bestückungsdruck und Stückliste zu bestücken.

Wir beginnen dabei mit den Tastern, die so einzusetzen sind, dass sie gleichmäßig plan auf der Platine aufliegen. Dem folgen die Elkos, die liegend montiert werden. Ihre Anschlüsse sind dazu zuvor um 90° abzuwinkeln und es ist die Polarität zu beachten (am Elko ist der Minusanschluss markiert, auf der Platine hingegen der Plusanschluss).

Als Nächstes sind die Niederspannungs-Hohlbuchse BU1 und das Kontrast-Einstell-Poti R41 zu bestücken und zu verlöten. Dabei ist auch bei BU1 auf plane



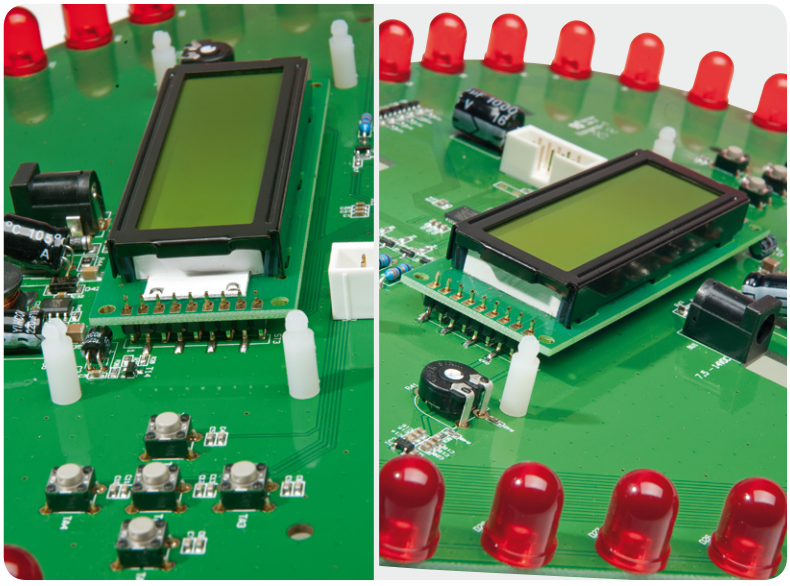


Bild 5: So ist das Display auf die Stiftleisten zu setzen. Hier ist auch das plane Aufsetzen der zum Bausatz angebotenen roten LEDs zu sehen.

Lage auf der Platine zu achten, und es ist reichlich Lötzinn zu verwenden, um später mechanische Kräfte durch den Steckeranschluss abzufangen.

Jetzt erfolgt das Einsetzen des Displays in die Stiftleisten. Dabei ist auf die seitenrichtige Lage zu achten: Mit der Herzspitze der Platine nach unten liegen die Anschlüsse für die Displayhinterleuchtung rechts, wie in Bild 5 zu sehen. Nach dem Verlöten der Anschlüsse folgt das plane und ebenfalls seitenrichtige Einsetzen (siehe Platinenfoto Bild 4) des Wannensteckers.

Eine Besonderheit bei diesem Bausatz ist die Möglichkeit, die LEDs mit den zugehörigen Vorwiderständen für das Blinkherz nach eigenem Wunsch zu bestücken (siehe „Elektronikwissen). Entsprechend sind aus dem mitgelieferten Widerstandssortiment die Widerstände R1, R4, R7, R10, R13, R16, R19 und R22 zu bestücken.

Bei der Bestückung der LEDs ist auf das polrichtige Einsetzen der LED zu achten. Der jeweils längere Anschluss der LED ist die Anode, er ist in das mit dem

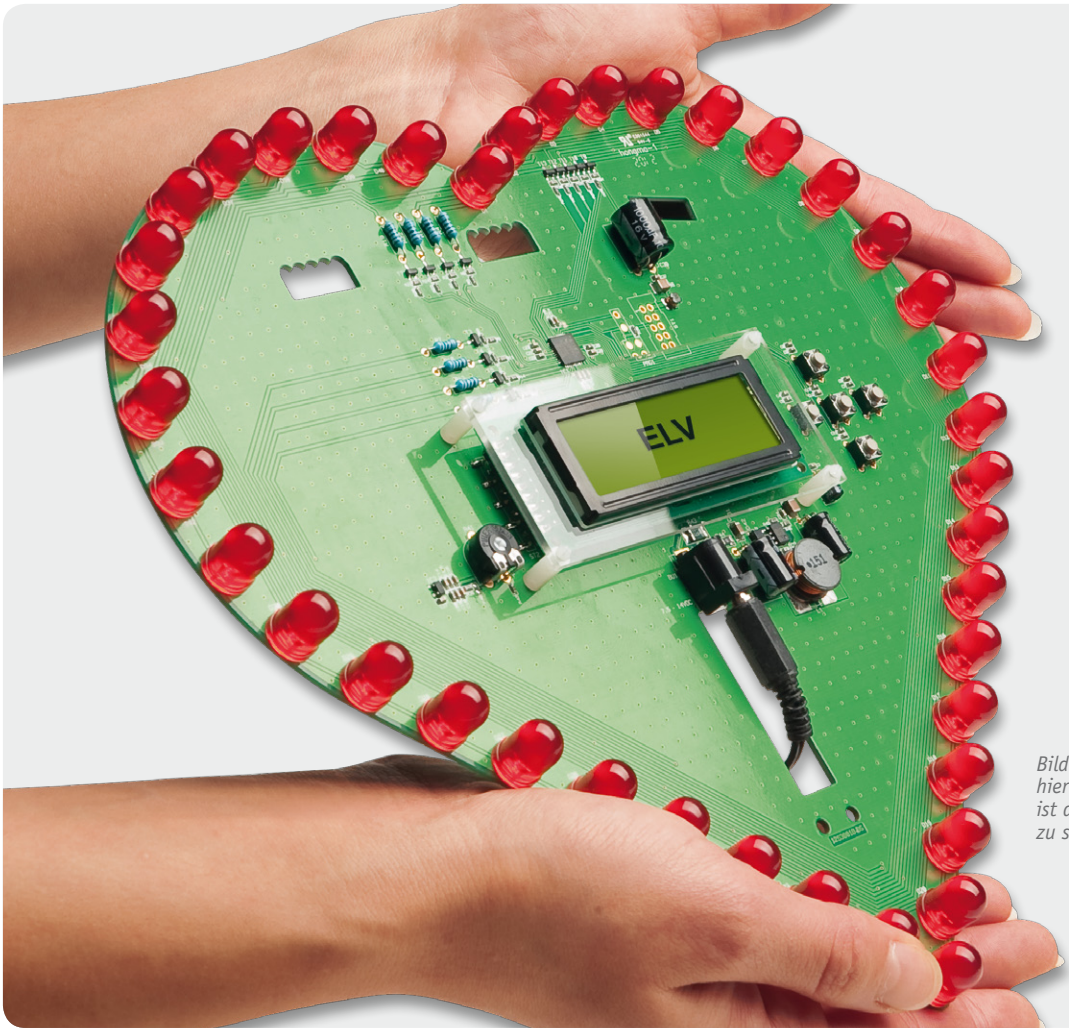


Bild 6: Das fertig aufgebaute Herz, hier bereits in Betrieb. In der Mitte ist die aufgesetzte Displayabdeckung zu sehen.

Pluszeichen markierte Loch zu führen. Der Abstand der LED zur Platine kann nach eigenem Wunsch gewählt werden, bei den als Zubehör angebotenen 10-mm-LEDs bietet sich, wie in Bild 5 zu sehen, das plane Aufsetzen der LEDs auf die Platine an.

Den Abschluss bildet schließlich das Einsetzen der vier Platinenabstandshalter in die vier Löcher um das Display herum und das Aufstecken der Abdeckscheibe auf die Abstandshalter. Die Abdeckung (Bild 6) ist unbedingt erforderlich, da es sich bei dem Display um ein ESD-empfindliches Bauteil handelt.

Ein Gehäuse für das Gesamtgerät wird nicht angeboten, da wohl jeder hier andere Vorstellungen haben wird. Man kann das Herz z. B. als Geschenk in einer passenden Pralinenschachtel präsentieren (Vorsicht, keine Metallfolien in der Schachtel lassen), in ein komplett selbst gestaltetes Gehäuse einbauen, eine passende Acryl-Abdeckscheibe davorsetzen, die Platine „nackt“ an die Wand hängen, für den Einsatz als AVR-Experimentierboard selbstklebende Standfüße auf die Platinenrückseite kleben, ganz nach Wunsch eben. Eine Reihe Befestigungslöcher und Aufhänge-Aussparungen sind bereits in die Platine eingearbeitet. Dem eigenen Gehäuse bzw. der Abdeckung kann man dann, wie erwähnt, auch den Abstand der LEDs zur Platine anpassen.

Atmel-Studio-Firmware

Die Firmware für das LBH40 steht im ELV-Web-Shop auf der Produktseite als Atmel-Studio-Projekt zum Download bereit. Die Firmware besteht dabei aus mehreren Komponenten, Bild 7 fasst diese zusammen.

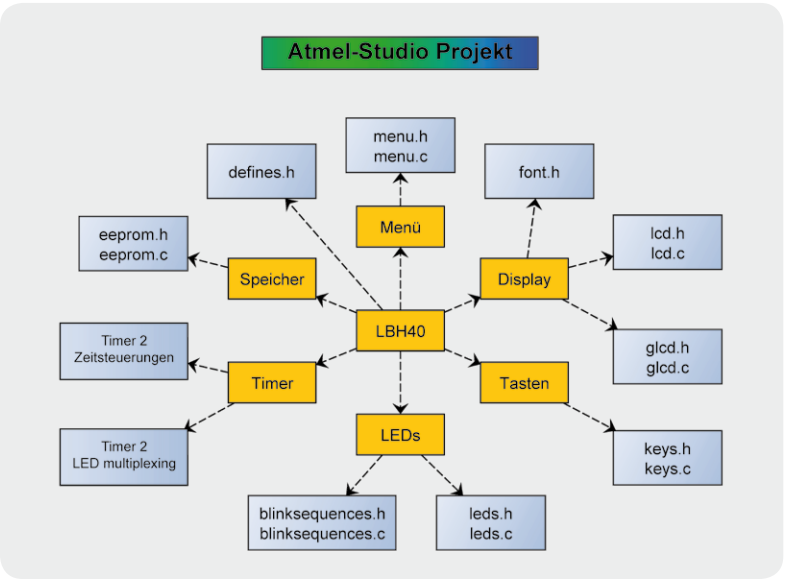


Bild 7: Die Komponenten der Firmware

Die Bedeutung der Komponenten:

- glcd = Grundroutinen zur Ansteuerung des grafischen Displays
- lcd = Texte auf dem Display darstellen
- eeprom = Speichern der Einstellungen
- leds = Grundfunktionen zur Ansteuerung der LEDs
- muster = Blinksequenzen
- keys = Tasten einlesen
- menu = Menüfunktionen

```
uint8_t blinksequences( uint8_t program_number )
{
  static uint8_t program_step_wait; // Wartezeit zwischen Programmschritten
  static uint16_t repeat = 0;      // Wiederholung des Programmschrittes
  static uint8_t program_step = 0; // Zähler Programmschritte

  if( program_step_wait != 0 )    // Wartezeit runterzählen
  {
    program_step_wait--;
  }
  if( program_step_wait == 0 )
  {
    switch( program_number )      // Programm ausführen
    {
    case PROGRAM_1:               // 1. Programm
      switch( program_step )      // Schritte ausführen
      {
         case PROGRAM_STEP_1:     // Schritt 1
           if( repeat == 0 )      // beim ersten Aufruf
           {
             repeat = 41;         // Anzahl Wiederholungen setzen
             LED_set_pattern( 0 );// Startmuster setzen
           }
           program_step_wait = 10;// Wartezeit setzen
           move_up_circle();      // Effekt ausführen
           break;
         case PROGRAM_STEP_2:     // Schritt 2
          if( repeat == 0 )       // beim ersten Aufruf
          {
            repeat = 0xfff8;      // Anzahl Wiederholungen
            LED_set_pattern( 0 ); // Startmuster setzen
          }
          program_step_wait = 10; //Wartezeit setzen

          if( collect_up() == 2)  //Effekt ausführen
          {
            repeat = 0;           //Ende collect_up
          }
          if ( (repeat&0x0f) == 0x08 ) //alle 9 Wiederholungen
          {
            LED_add_bit_to_pattern( 0 ); //LED hinzufügen
          }
          break;
        default:                  // Programm Ende
          program_step = 0;
          return true;
          break;
     }
     default:
       program_number = 0;
       return true;
       break;
    }
    if( repeat != 0 ) //Wiederholungen runterzählen
    {
      repeat--;
    }
    if( repeat == 0 ) //nächsten Programmschritt auswählen
    {
      program_step++;
    }
  }
  return false;
}
```

Bild 8: Der Beispielcode für die Erzeugung der Blinksequenzen

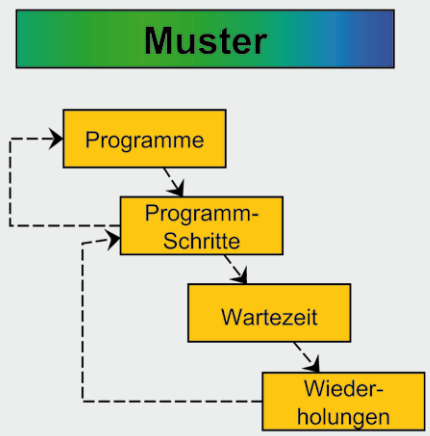


Bild 9: Der Aufbau der einzelnen Blinksequenzen als eigenes Programm

Stückliste

Widerstände:

0 Ω/SMD/1206	R44
10 Ω/SMD/0603	R38
47 Ω	R1, R4, R7, R10, R13, R16, R19, R22
120 Ω	R1, R4, R7, R10, R13, R16, R19, R22
270 Ω	R1, R4, R7, R10, R13, R16, R19, R22
820 Ω/SMD/0603	R42
1 kΩ/SMD/0603	R39
1,2 kΩ/SMD/0603	R2, R5, R8, R11, R14, R17, R20, R23, R25–R29
1,5 kΩ/SMD/0603	R45
2,2 kΩ/SMD/0603	R37
4,7 kΩ/SMD/0603	R46
10 kΩ/SMD/0603	R3, R6, R9, R12, R15, R18, R21, R24, R30–R36
PT10/liegend/1 kΩ	R41
Polyswitch/33 V/0,75 A/SMD/1812	R43

Kondensatoren:

1 nF/SMD/0603	C1, C4, C6, C8, C10, C12, C14, C19, C23
10 nF/SMD/0603	C5, C7, C9, C11, C13, C15, C20, C22, C25, C27, C29, C33
100 nF/SMD/0603	C3, C21, C24, C26, C28, C32, C34, C37
1 µF/SMD/0603	C16–C18
2,2 µF/SMD/1206	C31
10 µF/SMD/1210	C35, C38
10 µF/16 V	C2
220 µF/16 V	C36
220 µF/35 V	C30
1000 µF/16 V	C39

Halbleiter:

ELV121127/SMD	IC1
LM2675M-ADJ/SMD	IC2
TPS60400/SMD	IC3
BCW65C/SMD	T1–T8
BCW67C/SMD	T9–T14
SM4001/SMD	D41
SK14/SMD	D42
ESD9B5.0ST5G/SMD	D43

Sonstiges:

LC-Display DM12232-05YT mit Beleuchtung	LCD1
SMD-Induktivität, 150 µH/2,6 A	L1
SMD-Induktivität, 10 µH/0,9 A	L2
Hohlsteckerbuchse, 2,1 mm, print	BU1
Mini-Drucktaster, 1x ein, 1 mm Tastknopflänge	TA1-TA5
Wannen-Steckleiste, gerade, print, 2x 5-polig	ST1
2 Stiftleisten, 1x 9-polig, gerade, SMD	ST2, ST3
4 Platinenabstandshalter	
1 Displayscheibe	

- Timer 0 = Multiplexen der LEDs
- Timer 2 = Zeitsteuerung

Einige Funktionen müssen dabei zyklisch aufgerufen werden, dieses wird vom Hauptprogramm mit Hilfe des Timers 2 erledigt.

Display

Die GLCD-Routinen stellen nur die Grundfunktionen für das Grafikdisplay zur Verfügung, die weitergehende Ansteuerung zur Anzeige von Texten erfolgt in den LCD-Routinen. Dort werden die Texte aus dem Zeichensatz erstellt und das Durchlaufen der Texte erzeugt. Es stehen in der Komponente „font.h“ zwei Zeichensätze zur Verfügung, ein kleinerer mit 8 x 6 Pixel pro Zeichen, dieser wird z. B. für die Menü-Texte benutzt, und ein größerer mit 16 x 12 Pixel pro Zeichen zur Anzeige der gewünschten Lauftexte.

Auf das Einlesen der Tasten, den EEPROM und das Menü wird hier nicht näher eingegangen, dies sind Standardroutinen, die sich beim Analysieren der offen gelegten Firmware erschließen.

Blinksequenzen

Wie werden nun aber die einzelnen Blinksequenzen erzeugt? Dies soll hier anhand eines Beispiels (siehe Bild 8) erläutert werden.

Die unterschiedlichen Blinksequenzen werden im Folgenden als Programme bezeichnet, es stehen 16 Programme zur Verfügung.

Jedes Programm (Bild 9) kann mehrere Programm-Unterschritte aufweisen (implementiert sind bis 20 Unterschritte), wobei jeder Programm-Unterschritt wiederum mehrere Wiederholungen beinhalten kann. Die Pause zwischen den einzelnen Wiederholungen kann auch in jeder Wiederholung angepasst werden. So können sehr flexible Blinksequenzen erstellt werden mit variablen Wiederholungen, Effekten und Geschwindigkeiten. Die Funktion `blinksequences` wird zyklisch alle 10 ms vom Hauptprogramm aufgerufen.

Nur wenn die Wartezeit abgelaufen ist (das `program_step_wait` den Wert 0 erreicht hat), wird das als Parameter übergebene Programm bearbeitet.

Jedes Programm besteht aus einem oder mehreren Programmschritten, wobei in jedem Programmschritt beim ersten Durchlauf (`repeat = 0`) ein Startwert für das LED-Muster und die Anzahl der Wiederholungen gesetzt wird.

Danach wird die Wartezeit zwischen den Wiederholungen gesetzt und der gewünschte Effekt aufgerufen.

Bei jedem Funktionsdurchlauf wird die Anzahl der Wiederholungen vermindert, bis diese den Wert 0 erreicht, dann erfolgt die Auswahl des nächsten Programmschrittes.

Ist in einem Programm der letzte Programmschritt durchlaufen, werden die Programmschritte wieder zurückgesetzt und die Rückmeldung „Programm fertig“ an das Hauptprogramm ausgegeben.

Programmschritt 1 zeigt einen einfachen Ablauf, während Programmschritt 2 einen erweiterten Ablauf mit Hinzufügen weiterer LEDs zum Muster alle neun Wiederholungen zeigt. Weitere Programmbeispiele sind dem Quellcode zu entnehmen.

Effekte

Für die Effekte in einer Blinksequenz stehen mehrere Funktionen aus der Komponente „led.h“ zur Verfügung: Mit diesen vorgefertigten Effekten können LEDs im Kreis links/rechts herum oder von unten nach oben auf beiden Seiten gleichzeitig „wandern“, dabei entweder mit Ansammeln oben oder als Kreis mit Fortsetzung von unten. Man kann aber auch ganz eigene Muster kreieren.

Hier die im Programm vorhandenen Funktionen:

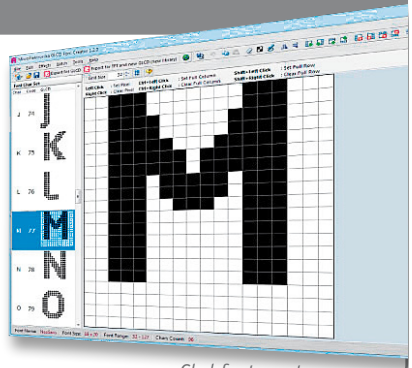
```
void move_left();           //schieben im Uhrzeigersinn
void move_right();          //schieben gegen den Uhrzeigersinn
void move_up();             //beide Seiten nach oben wandern
void move_down();           //beide Seiten nach unten wandern

void rotate_right();        //schieben im Kreis gegen den Uhrzeigersinn
void rotate_left();         //schieben im Kreis im Uhrzeigersinn
uint8_t collect_up();       //nach oben wandern mit Ansammeln
uint8_t collect_down();     //nach unten wandern mit Ansammeln

void move_up_circle();      //nach oben schieben im Kreis
void move_down_circle(); //nach unten schieben im Kreis

void invert();              //Abbild invertieren

void LED_set_pattern( uint64_t pattern );     //komplettes Bild setzen
void LED_add_bit_to_pattern( uint8_t bit ); //einzelne LED setzen
```

Die Ausgabe auf die LEDs erfolgt in einem Timer-Interrupt, dort werden die einzelnen LEDs einer Gruppe ausgewählt und die entsprechenden Gruppen aktiviert.

Zum Schluss noch ein Wort zur Gestaltung der Ausgaben auf dem Grafikdisplay. Der Zeichensatz enthält zwar bereits einige Symbole, jedoch ließen sich auch richtige Grafiken auf dem Display darstellen, dies wurde in dem Projekt bisher nicht berücksichtigt, so dass dort noch enormes Erweiterungspotential liegt. ELV

Hilfreiche Programme und Links:

Glcd font creator:	www.mikroe.com/eng/products/view/683/glcd-font-creator
Glcd bitmap converter:	http://en.radzio.dxp.pl/bitmap_converter
Atmel Studio:	www.atmel.com/tools/ATMELSTUDIO.aspx
Multiplexen	www.elektronik-kompendium.de/sites/kom/0211292.htm www.mikrocontroller.net/articles/LED-Matrix



Glcd font creator