Ralf Pongratz

7 Segment Basic Version

Reactive Lilght with 7 Segment Display

Also available as a kit at www.reaktivlicht.de

www.reaktivlicht.de

Grundversion
Reactive Light with 7 Segment Display

Ralf Pongratz

13. September 2013

Inhaltsverzeichnis

Teil I.
The circuit

1. Functional description

This circuit is the basic version of the reactive lights. It is easy to make with just a few elements and usable without parameterization. Instead of a simple LED a 7 segment LED display is used to show sequences of numbers and decimal points.

The measurement of brightness is done by a photo resistor (LDR). During daylight the circuit goes into a standby mode and is inactive. It just measures periodically the

brightness to detect the beginning of the night and set the circuit into an active mode. If in this mode the LDR is lit, the circuit will return the programmed sequence using the 7 segment LED display and wait for the next activation.

Because of the very small power consumption the circuit can be run for years with just a set of batteries.

2. Circuit diagram

Figure 1 shows the circuit diagram of the reactive light.

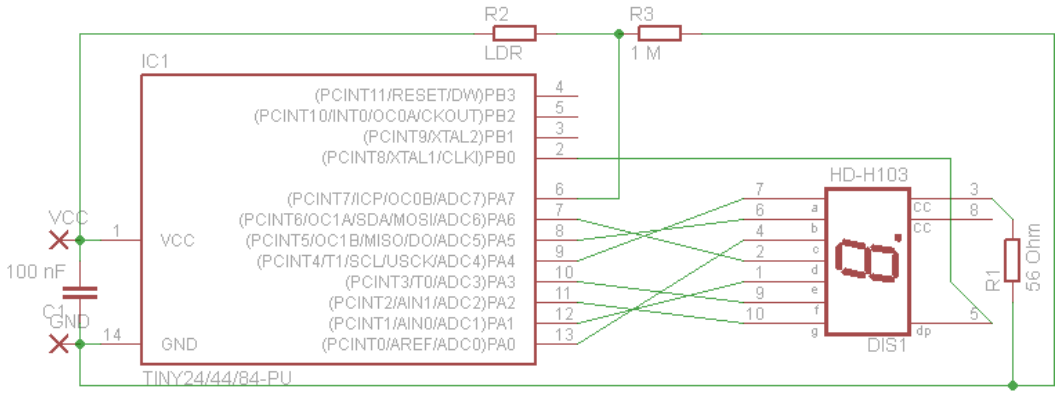

Abbildung 1: Circuit diagram.

On the left one can find the microcontroller, left of it the power supply. The circuit needs a voltage of 3 V. Two standard batteries in sequence make this voltage. The plus terminal must be connected to pin +, the minus terminal to the pin −. Above the IC is the measurement of the brightness, consisting of R3 and the photo resistor R2. On the right the 7 segment LED display (common cathode) with its resistor can be found. Because of the program structure just one resistor is needed for all eight LEDs of the display.

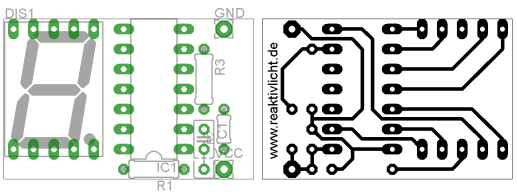

Abbildung 2: Layout of the circuit board.

A draft of the circuit board is shown in figure 2. The left figure shows the component side, the right one the circuit paths seen from the lower side. When mounting one has

to care about the polarity of the IC (the denting has to point to the capacitor).

3. Programmierung

```
$regfile = "ATtiny84.DAT"
$crystal = 16000
$hwstack = 3

Config Adc = Single , Prescaler = Auto
Ddra = &B01111111
Porta = &B00000000
Ddrb = &B00000001
Portb = &B00000000

Stop Ac

Wdtcr = &B11010000
Enable Interrupts

Const Threshold = 15
Const Daylight = 950
Const Maxdigits = 512

Dim Ldr As Integer
Dim Oldldr As Integer
Dim Deltaldr As Integer

Dim Daylightcounter As Integer

Dim E As Byte
Dim J As Integer
Dim X As Integer

Do

  Start Adc
  Ldr = Getadc(7)
  Stop Adc
  Deltaldr = Ldr - Oldldr
  Oldldr = Ldr

  If Deltaldr > Threshold Then
    Gosub Show
```

```
    Oldldr = 1024
  End If

  If Ldr > Daylight Then
    If Daylightcounter < 255 Then
      Daylightcounter = Daylightcounter + 1
    End If
  Else
    Daylightcounter = 0
  End If
  If Daylightcounter > 200 Then
    Gosub Pause_8
  End If

  Gosub Pause_0125

Loop

Show:
   X = 0
   E = 0
   While E < 11 And X < Maxdigits
     Readeeprom E , X
     If E < 11 Then
       For J = 0 To 100
         Gosub Show_dig
       Next J
       Gosub Pause_0125
     End If
     X = X + 1
   Wend
Return

Pause_0125:
  Wdtcr = &B11010011
  Reset Watchdog
  Powerdown
Return

Pause_8:
  Wdtcr = &B11110001
  Reset Watchdog
  Powerdown
Return
```

```
84
85  Show_dig:
86    If E = 0 Then
87      Gosub Dig_0
88    Elseif E = 1 Then
89      Gosub Dig_1
90    Elseif E = 2 Then
91      Gosub Dig_2
92    Elseif E = 3 Then
93      Gosub Dig_3
94    Elseif E = 4 Then
95      Gosub Dig_4
96    Elseif E = 5 Then
97      Gosub Dig_5
98    Elseif E = 6 Then
99      Gosub Dig_6
100   Elseif E = 7 Then
101     Gosub Dig_7
102   Elseif E = 8 Then
103     Gosub Dig_8
104   Elseif E = 9 Then
105     Gosub Dig_9
106   Elseif E = 10 Then
107     Gosub Dig_dot
108   End If
109 Return
110
111 Dig_dot:
112   Portb.0 = 1
113   Waitms 5
114   Portb.0 = 0
115   Waitms 5
116   Waitms 5
117   Waitms 5
118   Waitms 5
119   Waitms 5
120   Waitms 5
121 Return
122
123 Dig_0:
124   Porta.5 = 1
125   Waitms 5
126   Porta.5 = 0
127   Porta.0 = 1
```

```
128   Waitms 5
129   Porta.0 = 0
130   Porta.6 = 1
131   Waitms 5
132   Porta.6 = 0
133   Porta.1 = 1
134   Waitms 5
135   Porta.1 = 0
136   Porta.3 = 1
137   Waitms 5
138   Porta.3 = 0
139   Porta.4 = 1
140   Waitms 5
141   Porta.4 = 0
142   Waitms 5
143 Return
144
145 Dig_1:
146   Waitms 5
147   Porta.5 = 1
148   Waitms 5
149   Porta.5 = 0
150   Porta.0 = 1
151   Waitms 5
152   Porta.0 = 0
153   Waitms 5
154   Waitms 5
155   Waitms 5
156   Waitms 5
157 Return
158
159 Dig_2:
160   Porta.4 = 1
161   Waitms 5
162   Porta.4 = 0
163   Porta.5 = 1
164   Waitms 5
165   Porta.5 = 0
166   Waitms 5
167   Porta.6 = 1
168   Waitms 5
169   Porta.6 = 0
170   Porta.1 = 1
171   Waitms 5
```

```
172   Porta.1 = 0
173   Waitms 5
174   Porta.2 = 1
175   Waitms 5
176   Porta.2 = 0
177 Return
178
179 Dig_3:
180   Porta.4 = 1
181   Waitms 5
182   Porta.4 = 0
183   Porta.5 = 1
184   Waitms 5
185   Porta.5 = 0
186   Porta.0 = 1
187   Waitms 5
188   Porta.0 = 0
189   Porta.6 = 1
190   Waitms 5
191   Porta.6 = 0
192   Waitms 5
193   Waitms 5
194   Porta.2 = 1
195   Waitms 5
196   Porta.2 = 0
197 Return
198
199 Dig_4:
200   Waitms 5
201   Porta.5 = 1
202   Waitms 5
203   Porta.5 = 0
204   Porta.0 = 1
205   Waitms 5
206   Porta.0 = 0
207   Waitms 5
208   Waitms 5
209   Porta.3 = 1
210   Waitms 5
211   Porta.3 = 0
212   Porta.2 = 1
213   Waitms 5
214   Porta.2 = 0
215 Return
```

```
216
217 Dig_5:
218   Porta.4 = 1
219   Waitms 5
220   Porta.4 = 0
221   Waitms 5
222   Porta.0 = 1
223   Waitms 5
224   Porta.0 = 0
225   Porta.6 = 1
226   Waitms 5
227   Porta.6 = 0
228   Waitms 5
229   Porta.3 = 1
230   Waitms 5
231   Porta.3 = 0
232   Porta.2 = 1
233   Waitms 5
234   Porta.2 = 0
235 Return
236
237 Dig_6:
238   Porta.4 = 1
239   Waitms 5
240   Porta.4 = 0
241   Porta.3 = 1
242   Waitms 5
243   Porta.3 = 0
244   Porta.0 = 1
245   Waitms 5
246   Porta.0 = 0
247   Porta.6 = 1
248   Waitms 5
249   Porta.6 = 0
250   Porta.1 = 1
251   Waitms 5
252   Porta.1 = 0
253   Waitms 5
254   Porta.2 = 1
255   Waitms 5
256   Porta.2 = 0
257 Return
258
259 Dig_7:
```

```
260   Porta.4 = 1
261   Waitms 5
262   Porta.4 = 0
263   Porta.5 = 1
264   Waitms 5
265   Porta.5 = 0
266   Porta.0 = 1
267   Waitms 5
268   Porta.0 = 0
269   Waitms 5
270   Waitms 5
271   Waitms 5
272   Waitms 5
273 Return
274
275 Dig_8:
276   Porta.5 = 1
277   Waitms 5
278   Porta.5 = 0
279   Porta.0 = 1
280   Waitms 5
281   Porta.0 = 0
282   Porta.6 = 1
283   Waitms 5
284   Porta.6 = 0
285   Porta.1 = 1
286   Waitms 5
287   Porta.1 = 0
288   Porta.3 = 1
289   Waitms 5
290   Porta.3 = 0
291   Porta.4 = 1
292   Waitms 5
293   Porta.4 = 0
294   Porta.2 = 1
295   Waitms 5
296   Porta.2 = 0
297 Return
298
299 Dig_9:
300   Porta.4 = 1
301   Waitms 5
302   Porta.4 = 0
303   Porta.5 = 1
```

```
304     Waitms 5
305     Porta.5 = 0
306     Porta.0 = 1
307     Waitms 5
308     Porta.0 = 0
309     Porta.6 = 1
310     Waitms 5
311     Porta.6 = 0
312     Waitms 5
313     Porta.3 = 1
314     Waitms 5
315     Porta.3 = 0
316     Porta.2 = 1
317     Waitms 5
318     Porta.2 = 0
319 Return
320
321 End
```

In line 1 to 3 general settings are done. First the type of the processor is told to the compiler. Afterwards the frequency of the intern oscillator is set. At last the stack is set to 3 so that there is enough space for the variables of the program. The consequence is that the nesting depth of function calls can be maximum three, which is enough for this program.

Then the microcontroller is configured by the registers. First the analog digital converter that is used to measure the brightness is configured. Afterwards pins 0 - 6 of port A and pin 0 of Port B are configured as outputs and set to low. They are used for controlling the LED display. The other ports are inputs. In line 11 the analog comparator is switched off to reduce the power consumption. It is not used in this program. Line 13 and 14 configure the watchdog timer that is used for realizing the delays to 0.125 s and interrupt mode.

Now the contants and variables of the program are defined. `Treshold` is the minimum change of the brightness between two cycles that triggers the reactive light. By changing this value the sensitivity can be adjusted. If the brightness is above `Daylight`, the controller will go to standby mode. `Maxdigits` has to be set to the number of bytes of the EEPROM. This value depends on the used controller. The ATtiny22A has 128 Byte, the ATtiny44A 256 Byte and the ATtiny84A 512 Byte. The current brightness is stored in `Ldr`, the one of the previous cycle in `Oldldr`. `Deltaldr` is a variable that is used for storing the difference between these two values. In `Daylightcounter` the cycles with a brightness bigger than `Daylight` are counted. `E` contains the code of the digit to be shown. `J` is a counter. `X` is the adress of the EEPROM memory cell to be read.

The lines 30 to 56 contain the main program. At the beginning the analog digital converter is started in lines 32 to 41, the value of brightness read and the converter switched off again. The difference to the previous cycle is calculated and the actual

brightness stored at the variable `Oldldr` for the next cycle. If the difference is bigger than the treshold, the routine `Show` is called. Afterwards `Oldldr` is set to maximum value to prevent the program being triggered twice. That is possible when the brightness increases during showing the sequence. As of line 33 the conditions for the standby mode are checked. If the brightness is bigger that the daylight treshold, the daylight counter is incremented. Otherwise it is set to 0. If the daylight counter reaches the value 200, which means that during 200 cycles at 0.125 s daylight was detected, the procedure `Pause_8` is called. At last the controller is set to standby for 0.125 s. That's the interval for calculating the difference of the brightness between two cycles.

The procedure `Show` is in the lines 58 to 71. All bytes of the EEPROM are read until one byte is bigger than 10 or the end of the EEPROM is reached. For each value the procedure `Show_dig` is called 100 times. The time each digit is shown can be adjusted by the number of calls.

The lines 73 to 83 contain procedures to set the controller for 125 ms or 8 s to standby. At the beginning the watchdog timer is set to the specified time. After that the controller is set to standby. After the time is elapsed, the program is continued.

The procedure `Show_dig` in lines 85 to 109 calls depending on the variable `E` other procedures that show the corresponding digit at the display. These procedures are located in lines 111 to 319. Each procedure consists of an identical number of delay steps to make each digit the same length and brightness. One after the other the needed segments are lightened for one delay step. As always just one segment is lightened, a common resistor for all segments can be used.

The sequence to be shown has to be stored in the EEPROM. The values 0x00 - 0x09 represent the digits 0 - 9. The decimal point is coded as 0x0A. Values from 0x0B to 0xFF indicate the end of the sequence.

Teil II.
Known problems

4. Two segments of the display do not work.

During soldering probably a short-circuit between the pins of these two segments arose. It must be removed.

5. The device does not work and the microcontroller becomes very hot.

Either the microcontroller is placed incorrect ot the power supply is reverse poled.