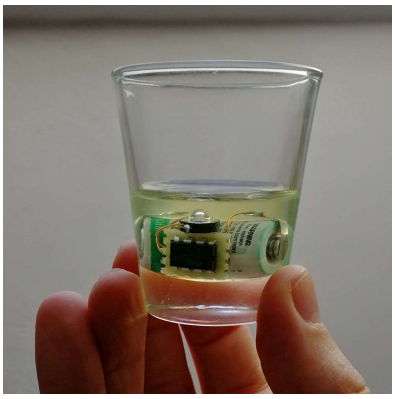
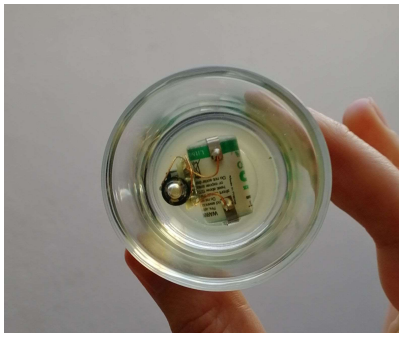
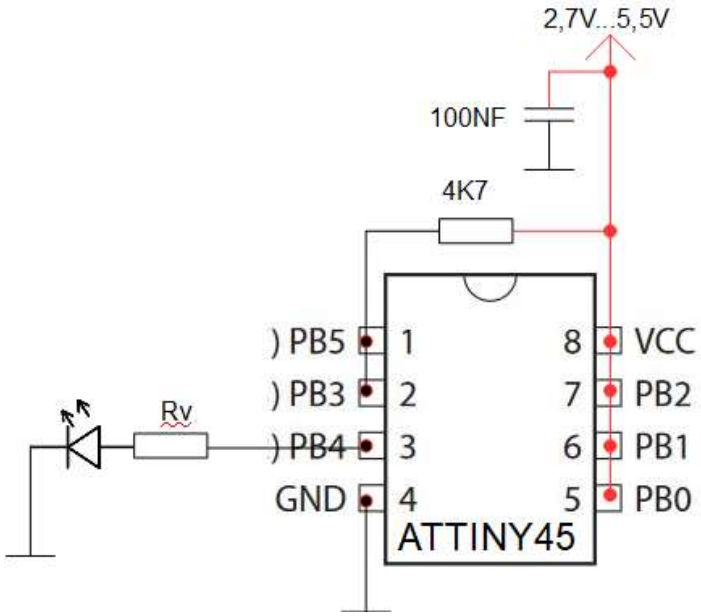
Blinkendes Schnapsglas
JO-Discovery 20_01_2021

Beim „blinkenden Schnapsglas“ handelt es sich um eine Schaltung, welche alle 2s einmal blinkt. Nach einem Jahr des Blinkens ändert sich die Abfolge etwas. Dann blinkt es zwei mal mit 2s Abstand und dann gibt es 4s pause bis sich die Abfolge wiederholt. So kann man quasi ablesen wie viele Jahre seit Beginn des Startens vergangen sind. Die Batterie sollte mehrere Jahre halten. Die durchschnittliche Stromaufnahme beträgt je nach LED und Vorwiderstand nur wenige uA, da sich der ATTINY fast nur im Power-Down-Modus befindet. Der Watchdog weckt ihn dann alle 2s einmal auf. Die LED blinkt ebenfalls nur sehr kurz (1ms). Die Schaltung wird mit Epoxidharz und einer kleinen Zelle (LS 14250) in ein Schnapsglas eingegossen.







Ausblick und Einschränkungen:

Die Zeitmessung ist äußerst ungenau und mehr als Schätzung zu betrachten. Wenn man wirklich will, dass die Zeitmessung auf den Tag genau funktioniert, sollte man einen kleinen RTC mit eingießen. Hierbei darauf achten, dass dieser nicht viel Strom verbraucht. Prinzipiell kann man einen RTC mit Sekunden-Takt nutzen um den ATTINY jede Sekunde einmal aufwachen zu lassen.

LED und Vorwiderstand:

die LED sollte mit wenig Strom möglichst hell leuchten. Es ist also ein Typ zu wählen, der hier passend ist. Der Vorwiderstand wird dann entsprechend ausgewählt.

```
/*
 * main.c
 * blinkendes_Schnapsglas V1.0
 * Created: 04.01.2021 12:01:06
 * Author : jo D
 */

// Achtung! Darauf achten, dass F_CPU=1000000UL definiert ist und die FUSES entsprechend gesetzt sind.

#include <avr/io.h>
#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <avr/power.h>
#include <avr/wdt.h>

#define LED_PIN PB4
#define LED_LEUCHTDAUER 1000 // Leuchtdauer in us
#define PULSES_PER_YEAR 15768000  // 31.536.000s pro Jahr

void LEDPin_init(void)
{
	DDRB |= (1 << LED_PIN);
	PORTB &= ~(1 << LED_PIN);
}

void resetWatchDog ()
{
	MCUSR = 0;
	WDTCR = (1<< WDCE ) | (1<< WDE ) | (1<< WDIF );
	WDTCR = (1<< WDIE ) |  (1<< WDP0 )| (1<< WDP1 )| (1<< WDP2 ); // 2s Sekunden TimeOut
	wdt_reset ();
}

// Schlafen und mit Watchdog interrupt wieder aufwachen
void sleep(void) {
	set_sleep_mode ( SLEEP_MODE_PWR_DOWN ); // sleep mode Power Down setzen
	cli();                                                                                                  // zur Vorsicht interrupts aus
	resetWatchDog ();		// reset WatchDog
	sleep_enable ();		// ermögliche sleep
	sei();                                                                                                  // interrupts ein
	sleep_cpu ();			// system geht in den sleep mode
	sleep_disable ();		// nachdem der interrupt geweckt hat, sleep_disable
}

void LED_AUS(void)
{
	PORTB &= ~(1 << LED_PIN);
}

void LED_EIN(void)
{
	PORTB |= (1 << LED_PIN);
}

int main(void)
{
	uint32_t Pulses = 0;
	uint8_t Jahre = 1;

	LEDPin_init();
	power_all_disable ();

	while (1)
	{
		if (Pulses >= PULSES_PER_YEAR)
		{
			Pulses = 0;
			Jahre++;
		}
		else
		{
			Pulses++;
		}

		if (Jahre > 1)
		{
			if (Pulses%(Jahre+1))
			{
				LED_EIN();
				_delay_us(LED_LEUCHTDAUER);
				LED_AUS();
			}
			else
			{
				_delay_us(LED_LEUCHTDAUER); // nur damit die Zeit in etwa stimmt. Ich bin sicher da gibts eine elegantere Lösung!
			}
		}
		else
		{
			LED_EIN();
			_delay_us(LED_LEUCHTDAUER);
			LED_AUS();
		}
		sleep();

	}
}

ISR(WDT_vect) {
	wdt_disable();  // bis zum nächsten mal wd disabled...
}
```