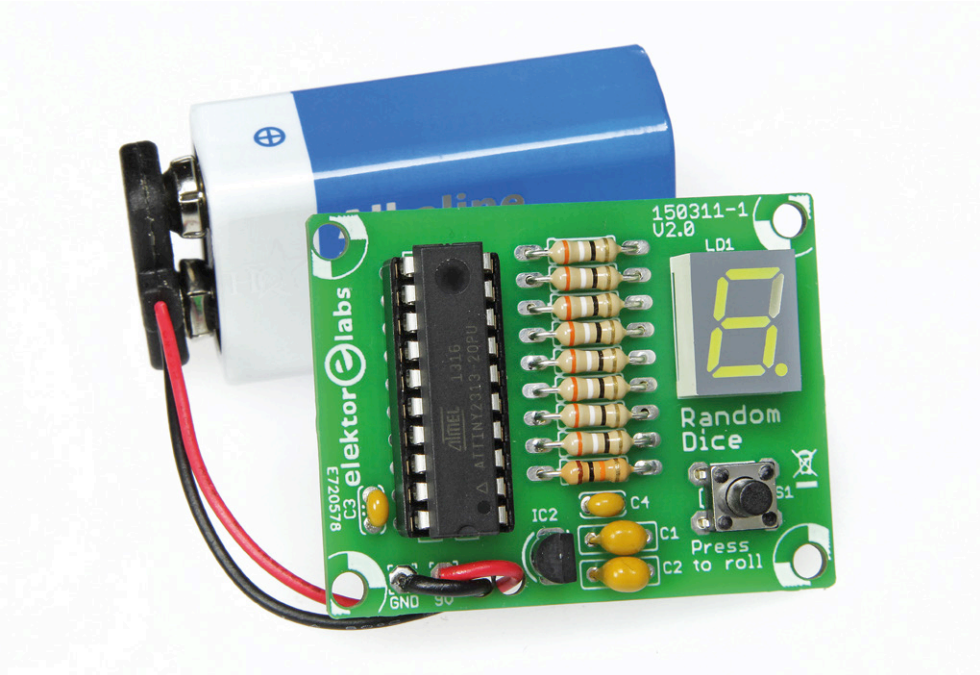
Tiny dobbelsteen

Elektronische dobbelsteen met de ATtiny2313



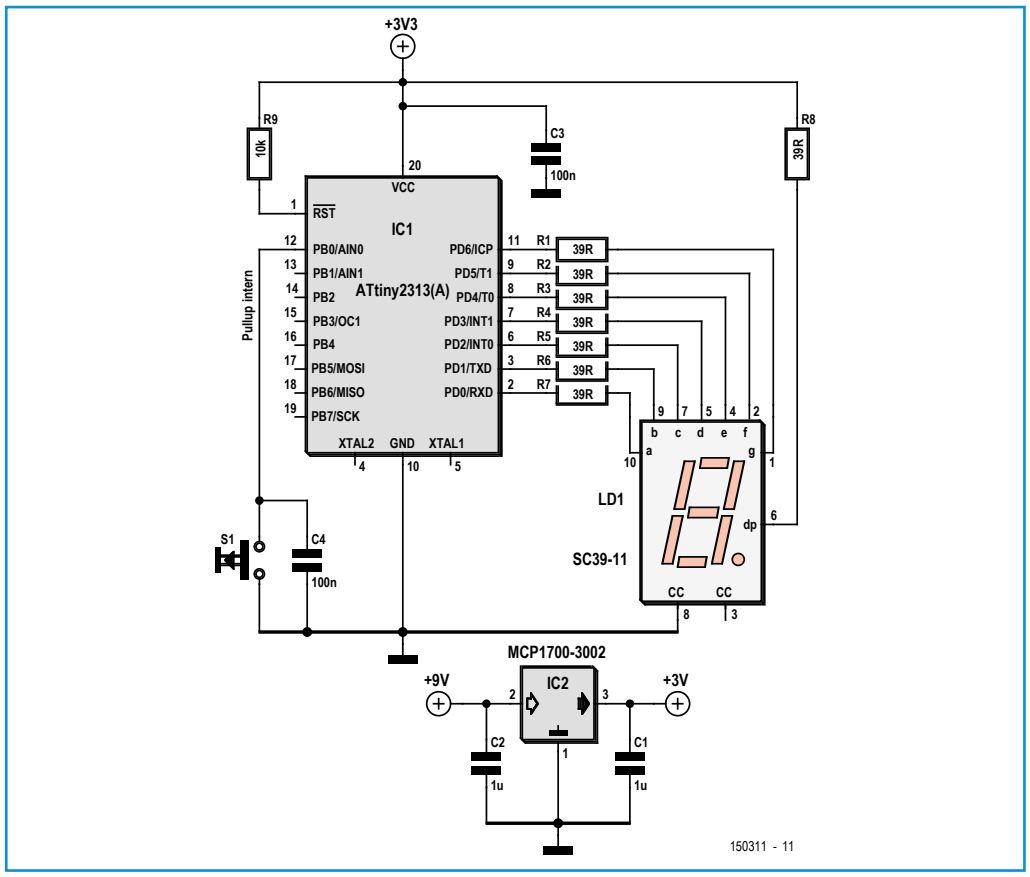
Florian Schäffer (Duitsland)

Deze eenvoudige elektronische dobbelsteen is een ideaal beginnersproject voor degenen die voor het eerst een soldeerbout gaan hanteren. Dit project biedt een ongevaarlijke kans om te oefenen met het bestukken van printen. De beloning voor het werk is een elektronische dobbelsteen, waarvan we niet eens de ogen hoeven te tellen, omdat het geworpen getal meteen decimaal wordt weergegeven. Comfortabeler kan het niet!

Onze schakeling simuleert een dobbelsteen. Bij het indrukken van een toets begint de dobbelsteen te rollen en geeft hij verschillende toevallige getallen tussen 1 en 6 weer; na een tijdje blijft één cijfer op het LED-display staan. Met het oog op de geschiktheid voor beginners zijn alle SMD's verbannen en zijn alleen bedrade componenten toegepast.

Dobbelsteen-schakeling

Zoals te zien is in **figuur 1** zijn een druktoets en een 7-segment LED-display verbonden met de I/O-pennen van een microcontroller van Atmel. De acht weerstanden R1...R8 begrenzen de stroom door de LED's. Een eenvoudige vaste spanningsregelaar maakt uit de batterijspanning van 9 V een stabiele 3 V voor de schakeling. C4 dient voor de contactdenderonderdrukking. Met de druktoets wordt de dobbelsteen geworpen. Als we de schakeling een tijdje niet gebruiken, wordt het display donker om de batterij te sparen. De LED van de decimale punt geeft aan dat de schakeling is ingeschakeld.
De dobbelsteen is in de Elektor-shop verkrijgbaar als bouwkit [1]. De geprogrammeerde controller wordt meegeleverd. De controller maakt gebruik van de geïntegreerde RC-oscillator op 8 MHz. In werkelijkheid is de klokfrequentie maar 1 MHz vanwege de interne deelfactor van 1:8. Een extern kristal voor de klokgenerator is dus niet nodig. Pullupweerstand R9 zorgt dat de hoogohmige reset-ingang van de microcontroller stevig 'hoog' (=inactief) blijft.



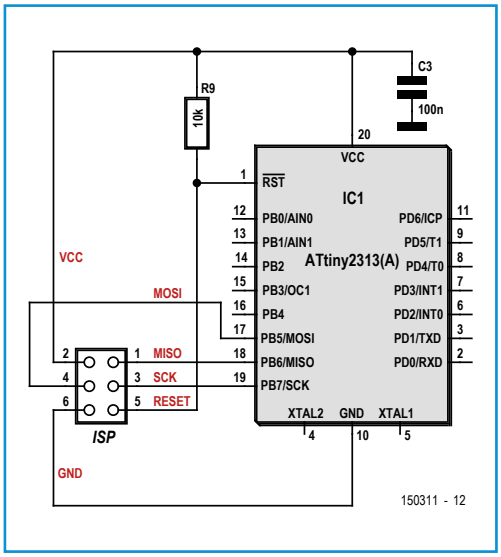
Figuur 1. Schema van de elektronische dobbelsteen met Atmel-microcontroller.

Broncode

Het dobbelsteen-programma in **listing 1** is gemaakt met de gratis toolchain WinAVR [2] en de bijbehorende GCC-compiler. Het is gecompileerd voor controllers van het type ATtiny2313A. De software werkt als volgt: Omdat een microcontroller geen echte toevalsgenerator bevat, wordt gebruik gemaakt van een snelle, permanent doorlopende teller (8-bits timer) om pseudo-willekeurige getallen te genereren.
Elke keer dat op de toets wordt gedrukt, worden twee interrupts gegenereerd (indrukken en loslaten). Bij elke tweede interrupt starten we een lus die (met steeds groter wordende tussentijden) getallen van 1...6 weergeeft en zo de dobbelstenen simuleert. Na het doorlopen van de lus wordt de stand van de continu op de achtergrond lopende teller afgevraagd en omgerekend naar een getal van 1...6 (rest bij delen door 6 bepalen en daarbij 1 optellen). Het resultaat wordt continu weergegeven: dit is de geworpen waarde. Omdat niemand weet wat de tellerstand is op het moment van indrukken en de teller te snel loopt om een gewenste waarde te 'raken', is valsspelen onmogelijk.
De aan te sturen segmenten voor de cijfers 0...9 liggen vast in de code; alleen de cijfers 1...6 worden gebruikt. Door het programma te veranderen zouden we ook andere cijfers of zelfs letters kunnen weergeven.

Programmeren

Zoals al eerder vermeld zit in de bouwkit een geprogrammeerde controller. Maar natuurlijk kunnen we deze ook herprogrammeren, bijvoorbeeld met een gewijzigd programma. De code wordt met een tekst-editor ingevoerd en dan vertaald naar machinetaal (gecompileerd). Het resulterende hex-bestand wordt met een geschikt programma zoals AVRDUDE [3] in het flash-geheugen van de microcontroller geschreven. Daar is nog wel wat hardware voor nodig. Naast een programmer, bijvoorbeeld de Atmel AVR-ISP MK2, is nog een kleine programmeer-adapter met IC-voet nodig. Die kunnen we snel even bouwen op een stukje gaatjesprint (zie **figuur 2**). Met deze hardware kunnen we het programma in de microcontroller steeds opnieuw overschrijven. Bij het programmeren van microcontrollers moeten we ook rekening houden met de fuses, waarmee de controller wordt geconfigureerd



Figuur 2. Kleine programmeer-adapter met zespolige ISP-aansluiting.

(bijvoorbeeld voor het inschakelen van de interne oscillator). Voor dit project gebruiken we gewoon de standaard-instellingen en hoeven we ons over de fuses niet te bekommeren.

De opbouw

Solderen: als de temperatuur van de soldeerbout instelbaar is, kiezen we 350...370 °C voor klassieke loodhoudende soldeer (wat voor zelfbouwprojecten heel handig is) of ongeveer 380...400 °C voor loodvrij soldeertin.

1. Plaats het onderdeel op de componentenzijde van de print. Zie de componentenopstelling bij de onderdelenlijst.
2. Verhit met de punt van de soldeerbout de aansluitdraad en het soldeervlak op de print ca. ½ seconde.
3. Voer soldeertin toe vanaf de andere kant.
4. Haal eerst het soldeertin en daarna de bout weg als de soldeer gesmolten is (na ca. 1 seconde).
5. Controleer de las.
6. Knip de overtollige lengte van de aansluitdraad af.

Volgorde van bestukken: Buig de aansluitingen van de onderdelen op maat met een platbektang. Buig niet te dicht bij de behuizing.

1. Weerstanden (richt de kleurringen uit van links naar rechts).
2. Condensatoren (let op de waarden).
3. Druktoets.
4. IC-voet (let op de markering voor pen 1).
5. 7-segment-display (let op de oriëntatie van de decimale punt).
6. IC2 (let op de oriëntatie). Druk het onderdeel niet te dicht op de print, maar laat ongeveer 5 mm ruimte tussen de print en de onderkant van de component. Anders treden grote buigkrachten op, die het onderdeel intern kunnen beschadigen.
7. Batterij-clip (voer de kabel van te voren door

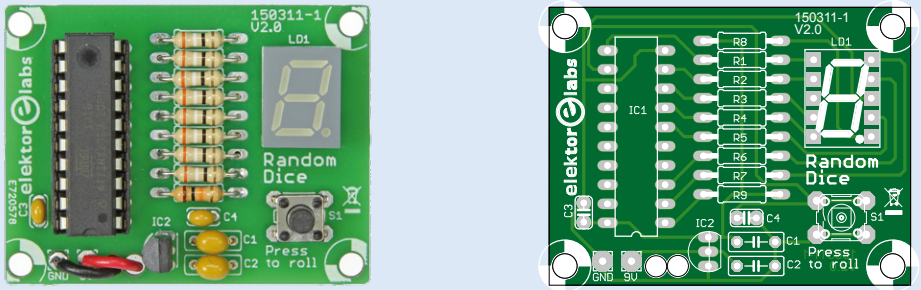
Onderdelenlijst

Weerstanden:
R1..R8 = 39 Ω, 5 %, ¼ W
R9 = 10 k

Condensatoren:
C1,C2 = 1 µ, folie, steek 5 mm
C3,C4 = 100 n, keramisch, steek 1/10“

Halfgeleiders:
IC1 = ATtiny2313A, geprogrammeerd
IC2 = MCP1700
LD1 = 7-segment LED-display SC39

Diversen:
S1 = druktoets met maakcontact, voor printmontage
Batterijclip voor 9 V-batterij
IC-voet voor IC1, 20-polig
Print 150311-1 Bouwpakket: 150311-71



Figuur 3. De volledig bestukte printplaat van de dobbelsteen.

de gaten voor de trekontlasting; polariteit: zwart = GND, rood = 9 V).

Eindcontrole: Controleer na het bestukken alle soldeerlassen. Zijn er aansluitingen vergeten? Zijn er koude lassen? Zijn er kortsluitingen door soldeerresten of draden? Controleer nogmaals de polariteit van de batterij en de oriëntatie van IC2. De print moet er na de opbouw uitzien zoals in **figuur 3**.
Sluit nu een batterij aan, zonder dat IC1 in het voetje zit. Nu moet de decimale punt oplichten. Als dat niet gebeurt, begint het favoriete werk van elektronici: foutzoeken. Als alles tot zover oké is, gaan we verder:
Verwijder de batterij.
Zet het geprogrammeerde IC1 voorzichtig in het voetje. Pen 1 (markering) wijst naar de batterijkabels.
Controleer of alle pennen zuiver boven de gaten van het voetje zitten. Als dat niet zo is, buig ze dan een beetje bij.
Druk het IC voorzichtig en gelijkmatig in het voetje met twee vingers en enige kracht, tot het goed op zijn plaats zit.
Sluit de batterij weer aan.
Druk op de druktoets... En daar gaan we!

Veel plezier met dit project!

(150311)

Weblinks

[1] Webpagina bij dit project: www.elektormagazine.nl/articles
[2] WinAVR: http://sourceforge.net/projects/winavr/
[3] AVRDUDE: www.nongnu.org/avrdude/

Listing 1.

```
#include <avr/io.h>
#include <util/delay.h>       // _delay_ms()
#include <avr/interrupt.h>    // IRQ

int main (void);

volatile uint8_t roll=0;
volatile uint16_t zeit=0;
const int8_t numbers [10] =   // 0..9: connection the ports binary to the segments. High active
{
/*      A
      F   B
        G
      E   C
        D         */
   0b00111111,    // 0
   0b00000110,    // 1
   0b01011011,    // 2
   0b01001111,    // 3
   0b01100110,    // 4
   0b01101101,    // 5
   0b01111101,    // 6
   0b00000111,    // 7
   0b01111111,    // 8
   0b01100111,    // 9
};

/**

   @brief   IRQ function will be called on IRQ @ PCINT. Make the dice rolling
*/
```

```
#if defined (__AVR_ATtiny2313__)
  ISR (PCINT_vect)
#elif defined (__AVR_ATtiny2313A__)
  ISR (PCINT_B_vect)
#else
  #error “Not supportet AVR”
#endif

{
   uint8_t i=0;
   roll++;        // how often the IQR was called? PCINT knows only toggle on key
                  // will be called twice each key pressed
                  // we need only one IRR to roll the dice
   if (roll == 2)
   {
      roll=0;                              // start again counting
      for (i=1; i < 40; i++)               // simulate rolling
      {
            PORTD = numbers[(i % 6)+1];    // Output. Counter Modulo 6 = 0-5  => +1 = 1-6
            _delay_ms(i*3);
      }
      PORTD = numbers[(TCNT0 % 6)+1];      // Output. Counter Modulo 6 = 0-5  => +1 = 1-6
      zeit=0;                              // Count time to switch off LED’s start again
   }
}

/**
   @brief   main
   @param   none
   @return  end state
*/
int main(void)
{
   PORTD = 0;                // PORTD complete off
   DDRD = 0xFF;            // PORTD complete as output
   DDRB &= ~(1 << DDB0);     // B0 Input
   PORTB |= (1 << PB0);    // Pull Up aktiv

   TCCR0B = (1 << CS02) | (1 << CS00); // 8 Bit Timer, Prescaler CLK/1024 => 1.000.000/256 = 3,9 kHz
                                       // => 0,000256 s/Impuls => x256 > all 0,065 s overflow
   GIMSK |= (1 << PCIE);          // PCIE IRQs enable
   PCMSK |= (1 << PCINT0);        // IRQ @ PB0 enable
   sei();                         // IRQs on

   while (1)  // endless
   {
      // Loop to switch off the display in x seconds
      for (zeit=0; zeit <=300; zeit++) // 300x100=30.000 ms = 30 seconds
         _delay_ms(100);

      PORTD = 0;  // all segments off
   }
   return 1;      // never
}
```