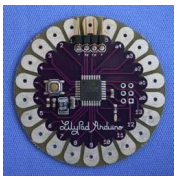


LilyPad

Experimente mit tragbarer Elektronik

Dr. Claus Kühnel

Die Integration von Elektronik in Bekleidung ist eine Ansatz, der neue Möglichkeiten multi-funktionaler, tragbarer Textilien zur Überwachung von Körperfunktionen, Bereitstellung von Kommunikationsmöglichkeiten, Datenübertragung, individueller Steuerungsfunktionen u.a.m. eröffnet. Modeartikel weisen Tastenfelder für Mobiltelefone und Anschlussmöglichkeiten für iPods oder MP3-Player auf. Mit Hilfe von Spezialkleidung ist man in der Lage Vitalfunktionen zu erfassen. Beim trainierenden Sportler kann die Herzfrequenz ebenso erfasst werden wie die Muskelarbeit. Risikopatienten können in Notfällen mit diesen Mittel gezielt Hilfe anfordern. Diese Aufzählung kann nahezu beliebig fortgesetzt werden.

Textilien, in die elektronische Systeme integriert sind, nennt man "Interactive Wear" oder „Wearable Electronics". Einen Überblick über die ständig wachsende Zahl von Angeboten ist in Funktionstextilien Online (siehe Links) zu finden.

Obwohl „Wearable Electronics“ immer wieder mal in den Mittelpunkt der Aufmerksamkeit rückt und zahlreiche Forschungsaktivitäten laufen, bleibt es um die Textilien selbst noch eher still. In diesem stark heterogen geprägten Umfeld gibt es noch viele technische Herausforderungen zu überwinden, bevor „Interactive Wear" nicht nur im Fashion- und Entertainment-, sondern auch im Gesundheits- und Pflegebereich zum Alltag gehören werden.

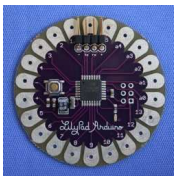
Spielerisch kann man sich aber bereits heute mit „Wearable Electronics“ auseinander setzen und dabei recht interessante Lösungen gestalten.

1. LiliyPad

LilyPad ist tragbare Elektronik, die von Leah Buechley entwickelt und gemeinsam mit der amerikanischen Firma Sparkfun in Komponenten umgesetzt wurde. Jede LilyPad Komponente wurde so entworfen, dass die Platinen keine Ecken aufweisen und mit deren großen Kontaktflächen in bzw. an die Kleidung genäht werden können. Außerdem sind alle Komponenten waschbar.

Leah Buechley ist Assistenz-Professor am MIT Media Lab, wo sie die High-Low Tech Forschungsgruppe leitet.

Um mit LilyPad zu starten, kann man das LilyPad Deluxe Kit (Abbildung 1) erwerben und hat damit alle heute verfügbaren Komponenten zur Hand: LilyPad Accelerometer, LilyPad Arduino Mainboard, LilyPad Bright White LED, LilyPad Button Board, LilyPad Buzzer, LilyPad Light Sensor, LilyPad Power Supply, LilyPad Temperature Sensor, LilyPad Tri-Color



LED, LilyPad Vibration Motor, FTDI Basic Breakout, Mini USB Cable zur Programmierung und nicht zuletzt eine Spule mit leitfähigem Garn.

Darüber hinaus benötigt man eine AAA-Batterie zur Spannungsversorgung und eine Nähnadel. Will man seinen Aufbau erstmal am Labortisch testen, dann tun es auch normale Drähte zur Verbindung der Komponenten.

Abbildung 1 LilyPad Deluxe Kit

Gesteuert werden die LilyPad Komponenten vom LilyPad Arduino Mainboard. Mit dem LilyPad Arduino Mainboard, zwei LilyPad Button Boards und 14 LilyPad Bright White LEDs kann beispielsweise der in Abbildung 2 gezeigte Fahrtrichtungsanzeige für Radfahrer aufgebaut werden. Die Spannungsversorgung kann über ein LilyPad Power Supply mit einer AAA-Batterie oder eine passende Lithiumbatterie erfolgen.

Abbildung 2 Fahrtrichtungsanzeige für Radfahrer

2. LilyPad Arduino Mainboard

Arduino ist eine Open-Source Plattform bestehend aus einem einfachen I/O Board, das ist in unserem Fall das LiLyPad Arduino Mainboard, und einer Processing/Wiring Entwicklungsumgebung (siehe Links).

Das LilyPad Arduino Mainboard kann wie alle anderen Arduino Controller zur Entwicklung von interaktiven Stand-Alone Objekten oder mit dem Computer verbundenen Anwendungen eingesetzt werden.

Abbildung 3 zeigt das LilyPad Arduino Mainboard. An den Pads sind die Atmel-konformen Pinbezeichnungen angetragen.

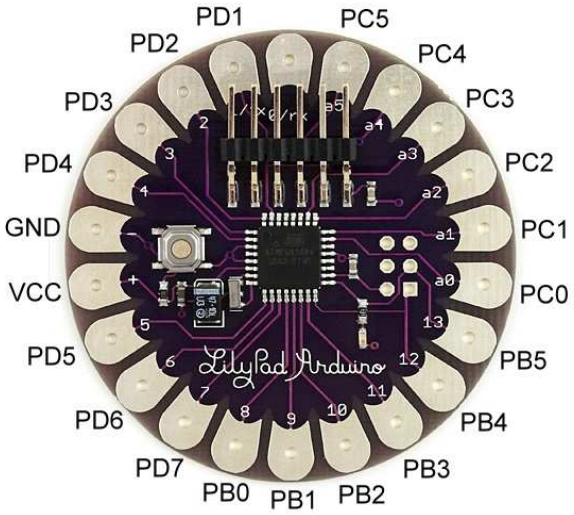


Abbildung 3 LiLyPad Arduino Mainboard

Die Arduino Entwicklungsumgebung (IDE) kann von http://www.arduino.cc herunter geladen werden. Arduino verwendet die GNU AVR-GCC Toolchain, Avrdude, AVR-LIBC und Code von Processing und Wiring. Die Arduino IDE läuft unter Microsoft Windows (XP & Vista), LINUX und Mac OS (Abbildung 6).

In der aktuellen Version 0012 (vom 18.09.2008) wurde Arduino an die neueren Versionen von AVR-GCC (4.3.0) und AVR-LIBC (1.6) angepasst, wodurch auch die aktuellen AVR-Mikrocontroller von Atmel unterstützt werden. Auf dem LilyPad Arduino Mainboard wird ein ATmega168V eingesetzt.

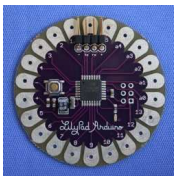
3. Anwendungsbeispiel „LightControlledBlinking“

Eine Sicherheits-Blinkschaltung, die beispielsweise auf der Jacke von Kindern angebracht werden kann, schaltet bei Dunkelheit ein und macht so auf den Passanten aufmerksam.

Neben dem LilyPad Arduino Mainboard werden zwei LilyPad Bright White LEDs für die Anzeigefunktion verwendet, die beispielsweise im Bereich von Schulter oder Arm beidseits montiert werden können. Der LilyPad Light Sensor misst das Umgebungslicht und schaltet bei Unterschreiten eines einstellbaren Helligkeitswerts die LEDs in den Blinkmode. Zur Spannungsversorgung dient das mit einer AAA-Batterie zu bestückende LilyPad Power Supply, welches auch den Hauptschalter trägt. Abbildung 4 zeigt die zur Blinkschaltung verdrahteten LilyPad Komponenten.

An das LilyPad Arduino Mainboard ist das FTDI Breakout, der RS-232/USB-Konverter, angesteckt, welches zum Programmdownload bzw. zur Kommunikation über USB mit dem PC verbunden werden kann.

Läuft das Programm zufrieden stellend und ist der gewünschte Schwellwert eingestellt, dann kann dieser Konverter abgezogen werden und die LilyPad Komponenten verrichten ihren Dienst autonom.



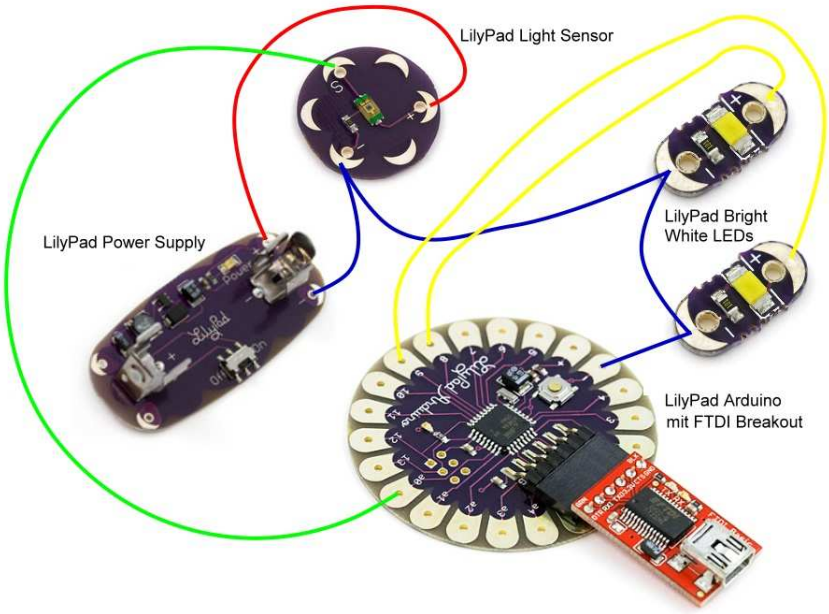


Abbildung 4 LilyPad Komponenten der Blinkschaltung

Das Programm, welches die LilyPad Komponenten zum Leben erweckt, kann in der Arduino Entwicklungsumgebung entwickelt und in Betrieb genommen werden. Der Entwicklungs-PC muss hierfür über USB mit dem Arduino Mainboard verbunden sein.

Der Quelltext kann im editiert, compiliert und schließlich in das LilyPad Arduino Mainboard herunter geladen werden. Abbildung 5 zeigt eine Quelltextausschnitt und den beendeten Download.

Das Programm belegt 3176 Bytes von den zur Verfügung stehenden 14336 Bytes. Der Rest des insgesamt 16 KBytes umfassenden Flash Memories wird durch den Bootloader belegt.

Wie Abbildung 7 zeigt, können im Monitor Mode der Arduino IDE die Ausgaben der seriellen Schnittstelle beobachtet werden.

In unserem Beispiel werden der gesetzte Schwellwert und der Wert für das gemessene Umgebungslicht ausgegeben. Liegt der Messwert unter dem Schwellwert, dann muss das Blinken aktiviert sein. Die zeichenweise Eingabe über die serielle Schnittstelle ist ebenso möglich, allerdings ist diese auf die druckbaren Zeichen beschränkt. Will man den Schwellwert in vollem Umfang ändern, dann muss man ein entsprechendes Terminalprogramm einsetzen.

Um abschließend noch einen Gesamteindruck von der Programmierung des LilyPad Arduino“ zu bekommen, zeigt Listing 1 den Quelltext des Anwendungsbeispiels „LightControlledBlinking.

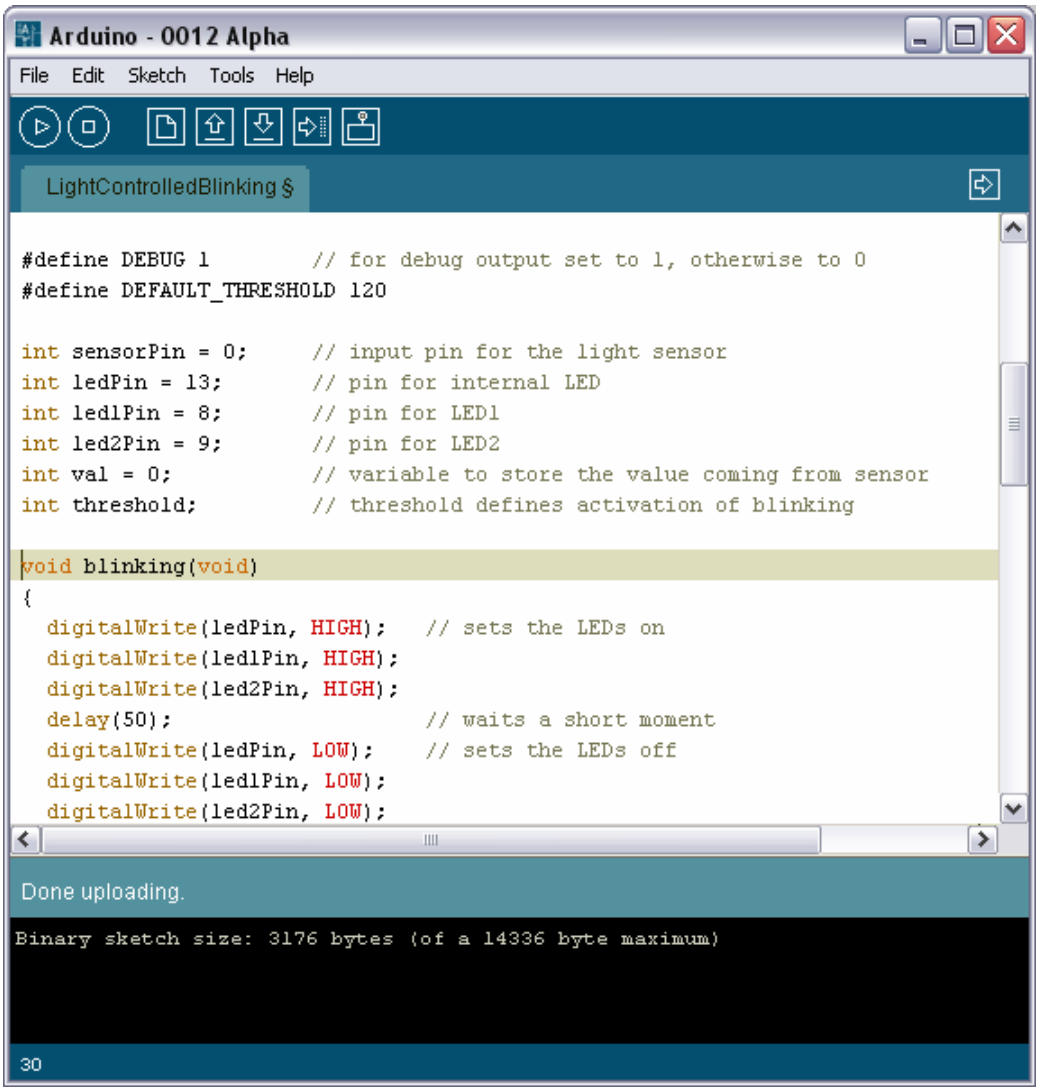


Abbildung 6 Quelltext „LightControlledBlinking“ in der Arduino IDE

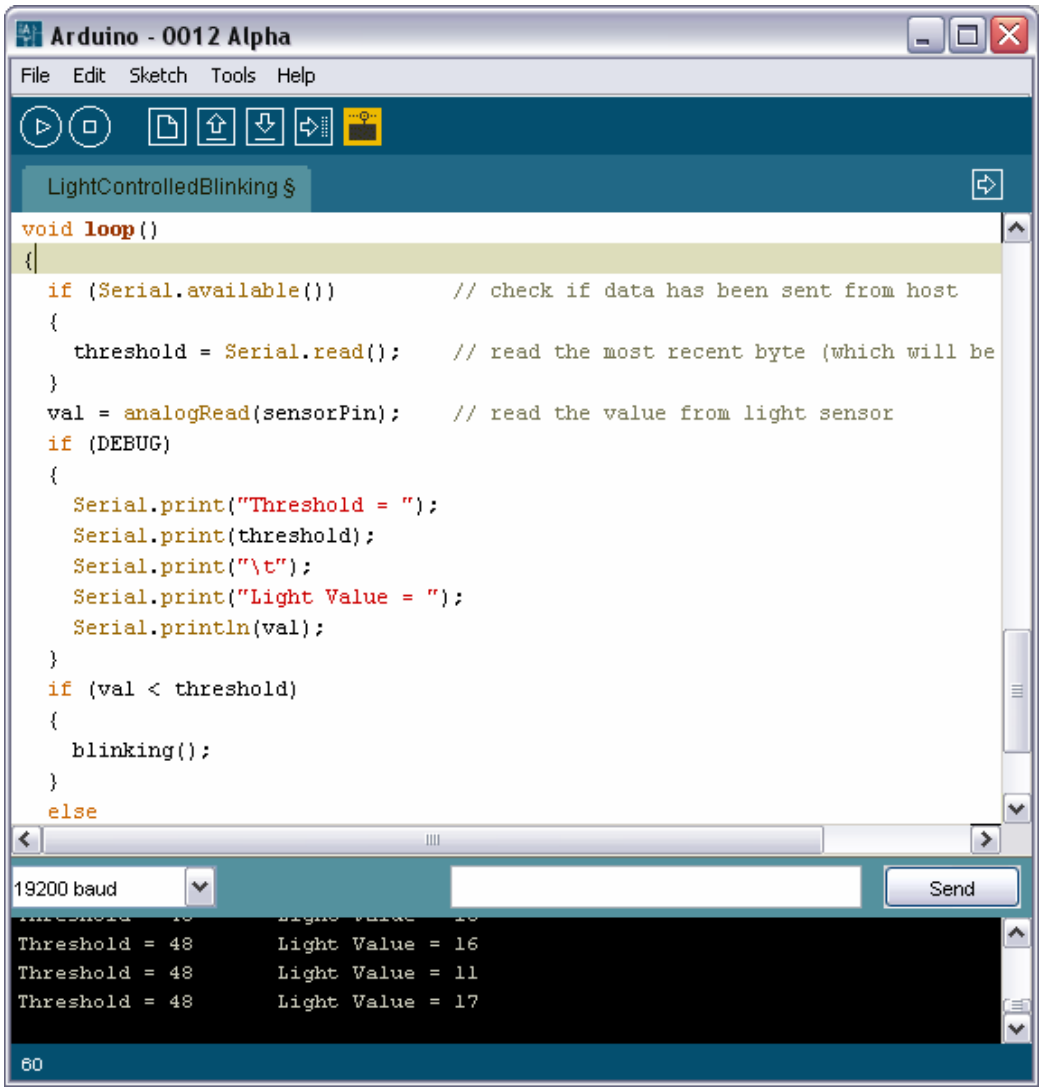
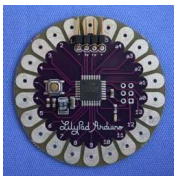
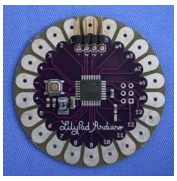


Abbildung 7 Monitoring Serial Output

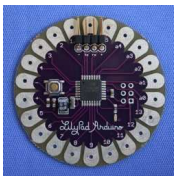


```
/*
 *name                      : LightControlledBlinking
 *programmed by             : Claus Kuhnel
 *date                      : 2008-12-07
 *micro                     : ATmega168V
 *tested with               : Arduino 0012 alpha
 */


/*
 * LightControlledBlinking
 *
 * Turns on and off light emitting diodes (LED) in blinking mode depending
 * of the amount of ambient light.
 * The switching point is defined by a constant as default and can be changed
 * by serial input a value between 0 and 255.
 *
 */

#define DEBUG 1        // for debug output set to 1, otherwise to 0
#define DEFAULT_THRESHOLD 120

int sensorPin = 0;     // input pin for the light sensor
int ledPin = 13;       // pin for internal LED
int led1Pin = 8;       // pin for LED1
int led2Pin = 9;       // pin for LED2
int val = 0;           // variable to store the value coming from sensor
int threshold;         // threshold defines activation of blinking

void blinking(void)
{
  digitalWrite(ledPin, HIGH);   // sets the LEDs on
  digitalWrite(led1Pin, HIGH);
  digitalWrite(led2Pin, HIGH);
  delay(50);                    // waits a short moment
  digitalWrite(ledPin, LOW);    // sets the LEDs off
  digitalWrite(led1Pin, LOW);
  digitalWrite(led2Pin, LOW);
  delay(100);                   // waits a short moment
  digitalWrite(ledPin, HIGH);   // sets the LEDs on
  digitalWrite(led1Pin, HIGH);
  digitalWrite(led2Pin, HIGH);
  delay(50);                    // waits a short moment
  digitalWrite(ledPin, LOW);    // sets the LEDs off
  digitalWrite(led1Pin, LOW);
  digitalWrite(led2Pin, LOW);
  delay(1000);                  // waits for a second
}

void setup()
{
  pinMode(ledPin, OUTPUT);    // declare ledPin as OUTPUT
  pinMode(led1Pin, OUTPUT);   // declare led1Pin as OUTPUT
  pinMode(led2Pin, OUTPUT);   // declare led2Pin as OUTPUT
  Serial.begin(19200);
  threshold = DEFAULT_THRESHOLD;
}
```



```
void loop()
{
  if (Serial.available())       // check if data has been sent from host
  {
    threshold = Serial.read();  // read the most recent byte (which will be from 0 to 255)
  }
  val = analogRead(sensorPin);  // read the value from light sensor

  if (DEBUG)
  {
    Serial.print("Threshold = ");
    Serial.print(threshold);
    Serial.print("\t");
    Serial.print("Light Value = ");
    Serial.println(val);
  }
  if (val < threshold)
  {
    blinking();
  }
  else
  {
    delay(1000);
  }
}
```

Listing 1 Quelltext Anwendungsbeispiel „LightControlledBlinking“



4. Links

Wearable Electronics in Funktionstextilien Online

http://www.funktionstextilien.de/content/view/488/122/

The NuMetrex Heart Rate Monitor System

http://www.numetrex.com/about/the-system

MIT Media Lab - leah buechley

http://web.media.mit.edu/~leah/

Sparkfun Electronics - Home | Development Tools | LilyPad

http://www.sparkfun.com/commerce/categories.php?c=135

Arduino - an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

http://www.arduino.cc

Processing - an open source programming language and environment for people who want to program images, animation, and interactions.

http://www.processing.org

Wiring - an open source programming environment and electronics i/o board for exploring the electronic arts, tangible media, teaching and learning computer programming and prototyping with electronics. It illustrates the concept of programming with electronics and the physical realm of hardware control which are necessary to explore physical interaction design and tangible media aspects.

http://wiring.org.co/

Physical Computing – Sensing and Controlling the Physical World

http://www.embedded.arch.ethz.ch/