An End-User Development Approach for Designing and Fabricating Interactive Paper

This work has been published at CHI 2015 - please cite Ramakers, R., Kashyap, T., and Luyten, K. Paperpulse: An integrated approach for embedding electronics in paper designs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Seoul, Korea, ACM (2015).

Raf Ramakers, Kashyap Todi, Kris Luyten
Hasselt University - tUL - iMinds
Expertise Centre for Digital Media
Diepenbeek, Belgium
firstname.lastname@uhasselt.be

Artifacts made using *PaperPulse*: (a) An interactive game (b) A birthday invitation card with a code-slot to reveal the date (c) A diet-card to keep track of daily eating habits.

In *Workshop on End User Development in the Internet of Things Era* (CHI2015 workshop). April 19, 2015. Seoul, Korea.

Abstract

This paper covers *PaperPulse* [3], a design and fabrication approach that enables designers to produce standalone interactive paper artifacts by augmenting them with electronics. With PaperPulse, users overlay visual designs with widgets provided in the design tool. PaperPulse provides different families of widgets, designed for smooth integration with paper, for a total of 20 different interactive components. We also contribute a demonstration and recording approach, *Pulsation*, that allows specifying interaction logic. Using the final design and the recorded Pulsation logic, PaperPulse generates layered electronic circuit designs, and code that can be deployed on a microcontroller. By following automatically generated assembly instructions, designers can seamlessly integrate the microcontroller and widgets in the final paper artifact.

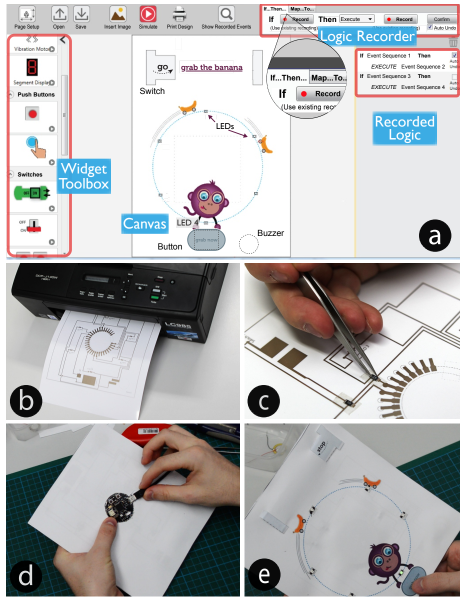

Figure 1: The *PaperPulse* workflow. (a) Design and specify logic; (b) print sheets; (c) upload generated program to microcontroller; (d) assemble; (e) final paper artifact.

Introduction

There has been a growing interest in different fields and communities (e.g. HCI research, maker movement, engineering and marketing) in making paper interactive by augmenting it with electronics. It has become possible to produce low-cost paper versions of PCBs in lab environments [1], and bring liveness to paper artifacts such as books and posters [2]. Although advancements in fabrication tools for electronic circuits, such as conductive pens, threads, inkjet printers [1] and vinyl cutters [4], make building paper circuits accessible to people, the requirement of possessing elementary knowledge of electronics and programming can make creating interactive paper circuits a challenging task.

To enable end-users to augment paper designs with electronics, we introduce *PaperPulse* [3], a design tool that assists users and automates parts of the design, programming, and fabrication process for electronic paper circuits. With PaperPulse, users can make stand-alone interactive paper artifacts, in which electronic components are seamlessly integrated in visual designs, without requiring skills in electronics or programming. PaperPulse has been validated with designers who have little to no background in electronics or programming. As such they represent a wide diversity of possible end-users that can use PaperPulse for their creative needs.

Walkthrough: The Hungry Monkey Game

We start with a walkthrough that illustrates the process of designing and fabricating a paper game with PaperPulse (Figure 1). The game consists of a loop of six LEDs that consecutively turn on and off. The objective of the game is to “grab the banana” by pressing a button at the moment when a particular LED lights up. A buzzer rings for a short duration each time the player succeeds.

Step 1: Designing the Interactive Paper Layout

The user starts by specifying the dimensions of the paper design, importing pre-designed images, and arranging them onto the canvas (Figure 1). Next, the user overlays this design with interactive components (six LEDs, a buzzer, a pull switch, and a button), all readily available in the widget toolbox.

Step 2: Defining and Verifying Logic Iteratively

The user first links the “stop–go” switch to the circle of the LEDs. When the switch is turned on, the LEDs blink in a looping pattern. Figure 2 illustrates how recordings are created by demonstrating actions specific to widgets,

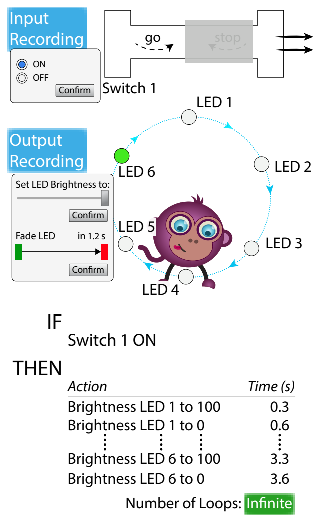

Figure 2: Recording an *if–then* rule for the LED sequence when a switch is turned on.

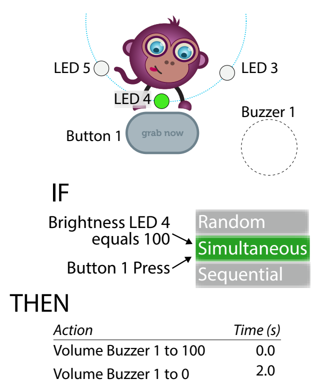

Figure 3: Recording another rule to ring a buzzer if the button is pressed at the moment "LED 4" turns on.

and the specified logic for the desired behavior.

To verify the recorded rule, the build-in *simulator* allows to interact with the virtual widgets through the PaperPulse tool and observe the corresponding output.

Next, the user records the logic for the "grab now" button. If pressed at the moment when the LED under the monkey ("LED 4") lights up, the buzzer should ring to indicate that the game is completed. Figure 3 illustrates the specified logic to obtain this behavior.

Step 3: Printing and Assembly
The printing process starts by generating: (1) An electronic circuit that connects widgets to pins on the microcontroller[1]. (2) PDF files consisting of the electronic circuits, widget-specific assembly lines (e.g. cut lines, fold lines), and visual designs. (3) Microcontroller code. (4) A customized tutorial to guide the user through the whole fabrication process. As shown in Figure 1e, the resulting end-product, powered by a battery, can now be used as a standalone paper game.

Pulsation: Specifying Sensor Logic By Demonstration

Pulsation allows users to specify logic by demonstrating and recording actions using only the visual design elements. To provide more fine-grained control than is possible with demonstration alone, Pulsation augments widgets and the demonstrated actions with dialogs that allow fine-tuning specific properties.

Logic is defined by specifying relationships between *input sets* (conditions that have to be fulfilled) and *output sets* (consist of one or more output actions). These relationships can be specified using either *if–then* or *map–to* rules. For if–then rules, the output set is executed when all conditions in the input set have been satisfied. The walkthrough (Figure 2, 3) shows two examples of such rules. For map–to rules, parameters of *input set* (e.g. raw value, repetitions, progress within the set) are continuously mapped to parameters of the *output set* (e.g. value, progress of actions, speed with which the actions are executed). An Example usage of a map–to rule is mapping the value of a slider to the brightness (value) of LEDs.

Input and Output Sets
Input sets specify conditions that have to be fulfilled and thus consist of one or more conditions related to input or output widgets. Different types of conditions are supported by Pulsation: *momentary conditions* (true for a brief amount of time), *discrete conditions* and *continuous range* conditions (current value within specified range). Simple patterns of conditions can be specified using timing and matching options supported by input sets. These options will be demonstrated during the workshop. Figure 3 gives an example of an input set that is fulfilled when a push button is pressed at the same time that an LED lights up.

Output sets consist of one or more output actions. Pulsation supports two types of output actions: (1) *Discrete output actions*, such as lighting up an LED or setting the digit of a seven-segment display, and (2) *Range output actions* specify an output range that has be transitioned. An optional time parameter can be specified by the user. Examples include, fading an LED in or out or realizing a count-down or count-up with a seven-segment display.

As was shown in Figure 2, output sets allow to specify

[1] Threadneedle from modlab.co.uk

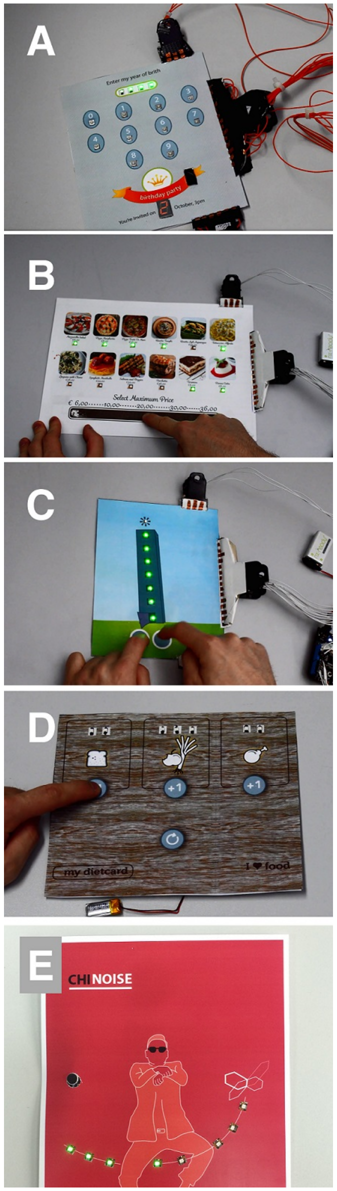

Figure 4: Example interactive paper artifacts created with PaperPulse.

delays between recorded actions. The *loop* construct offers the possibility to execute the set of actions multiple times.

If–then Rules
One way to relate input to output sets with Pulsation is using if–then rules. These rules allow to *execute* or *stop/reset* an output set when all conditions of an input set are met. OR-relations are indirectly supported using multiple if–then rules. An existing output set can also serve as input set for another if–then rule, thus allowing for nested rules. When input sets solely consist of stateful conditions, i.e. *discrete state conditions* and *continuous range actions*, it is often desirable to undo all actions performed in the output set once the conditions in the input set are not fulfilled. Pulsation automatically infers for every if–then rule whether this undo is appropriate (i.e. if the input set consist of only stateful conditions) and will then suggest to automatically undo all state changes caused by this rule when the input set is not fulfilled anymore.

Map–to Rules
Map–to rules allow for linear mapping of a derived parameter of the input set to another parameter of the output set. For example, mapping the speed with which a push button is continuously tapped to the number of LEDs that light up. Pulsation supports various derived parameters, such as *Value*, *Progress*, *Repetition*, *Time* and *Speed*.

Artifacts created with PaperPulse
Using Pulsation and widgets offered by PaperPulse, users can interactively realise a broad range of interactive paper applications as shown in Figure 4. (a) A special birthday card for which the user needs to enter the year of birth before the date of the party is revealed. (b) An interactive restaurant menu to quickly filter through menu options using a maximum price. (c) A tapping game: the faster you tap the two buttons, the more LEDs in the tower light up. (d) A diet card to track your daily food consumption. (e) An interactive poster: the higher the volume, the more LEDs will light up.

Conclusion
As illustrated in the walkthrough, PaperPulse streamlines the entire design and fabrication process, assisting and supporting end-users during each step of creating interactive paper. With PaperPulse, end-users can create stand-alone paper artifacts without requiring knowledge of electronics or programming. We believe electronic components used in PaperPulse will soon become cheap enough to enrich every paper design, from books, posters, and business cards to ephemeral packaging material and flyers.

References
[1] Kawahara, Y., Hodges, S., Cook, et al. Instant inkjet circuits: Lab-based inkjet printing to support rapid prototyping of ubicomp devices. In *Proc. UbiComp '13*,

[2] Qi, J., and Buechley, L. Electronic popables: Exploring paper-based computing through an interactive pop-up book. In *Proc. TEI '10*,

[3] Ramakers, R., Kashyap, T., and Luyten, K. Paperpulse: An integrated approach for embedding electronics in paper designs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Seoul, Korea, ACM (2015).

[4] Savage, V., Zhang, X., and Hartmann, B. Midas: Fabricating custom capacitive touch sensors to prototype interactive objects. In *Proc. UIST '12*,