Hasselt University

# SoftMod: A Soft Modular Plug-and-Play Kit for Prototyping Electronic Systems


*Author:*
Mannu LAMBRICHTS
Master Computer Science

*Supervisor:*
Prof. dr. Raf RAMAKERS

*Co-Supervisor:*
Mr. Jose Maria TIJERINA MUNOZ


*Master thesis in fulfillment of the requirements for the degree of master computer science*

August 28, 2019

# *Abstract*

We present SoftMod, a new modular electronics kit consisting of soft and flexible modules that magnetically snap together. Unlike existing modular kits, SoftMod tracks the topology of interconnected modules and supports basic plug-and-play behavior as well as advanced user-specified behavior. As such, the shape of a SoftMod assembly does not depend on the desired behavior, and various 2D and 3D electronic systems can be realized. While the plug-and-play nature of our modules stimulates play, the advanced features for specifying behavior and making a variety of soft and flexible shapes, offer a high-ceiling when experimenting with novel types of interfaces, such as wearables, and interactive skin and textiles. SoftMod assemblies can also be formed into rigid shapes to create high-fidelity prototypes. The fabrication of the toolkit is fully DIY compatible as it only adapts existing techniques and creates a new one.

# *Acknowledgments*

In the first place, I would like to express my gratefulness to Prof. Dr. Raf Ramakers and Mr. Jose Maria Tijerina Munoz for guiding me throughout the entire process of this thesis. Without their guidance and assistance, it would not have been possible to accomplish the goals set for this thesis. Secondly, I would like to thank my girlfriend for always supporting me whenever I need it. My friends also deserve a special mention for all long-winded discussions we had about my thesis. At last, I would like to thank my parents and brother for their motivation and everyone who has proofread this thesis because, without you, this thesis text would still be full of spelling mistakes.

Thank you!

# *Dutch Summary*

Dit hoofdstuk bevat een Nederlandstalige samenvatting van deze thesis. Om te beginnen zal de eerste sectie een introductie op de thesis geven. De volgende sectie beschrijft het algemene concept, samen met het gedrag en een aantal use cases. De secties daarop beschrijven het design van SoftMod, zoals elektronische componenten en hardware designs, het design van de flexibele PCB's, de belangrijkste algoritmes en een techniek om van een SoftMod schakeling over te gaan tot een high-fidelity prototype. Tot slot zal de laatste sectie de conclusie van deze thesis geven samen met een discussie over limitaties.

## Introductie

Deze thesis presenteert een nieuwe manier van prototypen voor niet-elektronica ingenieurs. SoftMod (Soft Modules) is een flexibele modulaire elektronische toolkit die toelaat elektronische schakelingen te maken door flexibele modules aan elkaar te klikken, zoals in afbeelding 1. SoftMod bevat modules voor sensoren, output, logica, voeding en draadloze communicatie. In tegenstelling tot bestaande plug-en-play modulaire kits zoals LittleBits [13] en MakerWear modules [36] kan de topologie van een SoftMod schakeling bepaald worden waarop het standaard gedrag aangepast wordt. Meer geavanceerde logica kan geprogrammeerd worden door de SoftMod IDE, die ook de topologie visualiseert. SoftMod bevat ook uitrekbare kabels die het mogelijk maken grotere 2D of zelfs 3D structuren te maken. SoftMod stimuleert designers om nieuwe interactie systemen te onderzoeken, zoals bijvoorbeeld wearables en interactieve textiel en huid. Daarnaast trekt SoftMod ook kinderen en volwassenen aan om met sensoren en elektronica te spelen zonder risico op schade. SoftMod is naast deze thesis ook geüpload voor TEI 2020 ([38]).

SoftMod is volledig DIY compatibel en bestaat uit modules die zeer flexibele PCB's, zoals in figuur 2a, bevatten. Deze PCB's zijn geproduceerd door gebruik te maken van een nieuwe techniek die lagen van koper en Kapton tape op elkaar stapelt. De sleutel tot deze techniek is de combinatie van een CO2 en fiber laser. Door een van de twee lasers te gebruiken kan men respectievelijk ofwel Kapton ofwel koper snijden. Daarnaast moeten de magneten in de connectoren zeer precies gemonteerd worden, dit wordt gedaan door gebruik te maken van houders.

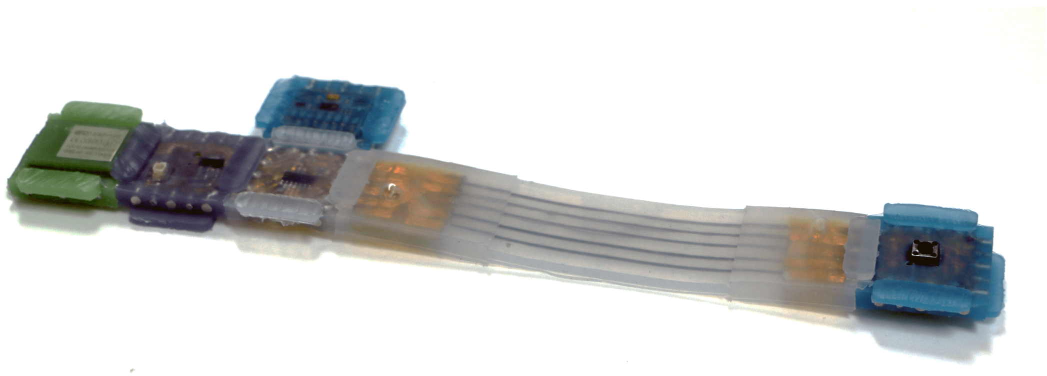
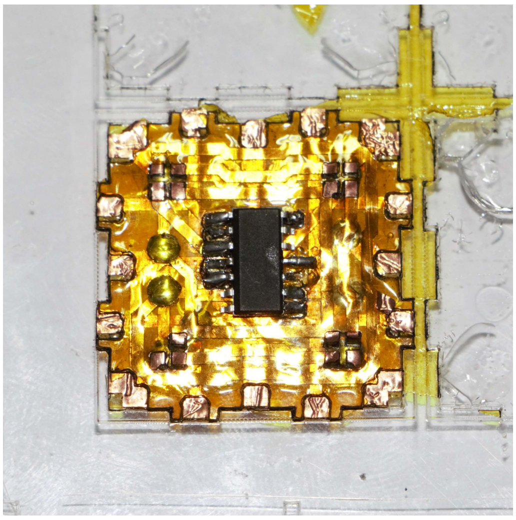


FIGURE 1: Een SoftMod schakeling.



(A) Een DIY flexibele PCB.

(B) Software tool die gebruikt wordt om modules in 3D om te zetten naar 2D.

Tot slot kan SoftMod gebruikt worden om snelle en eenvoudige prototypes om te zetten naar high-fidelity prototypes. Hiervoor bevat SoftMod een tool die toelaat om bepaalde SoftMod modules te plaatsen op specifieke plaatsen in een 3D model. Hiervan kan dan een 2D circuit gegenereerd worden zodat de modules precies op de gewenste plaats zitten. Deze techniek kan gebruikt worden indien een schakeling rond een bepaald object moet gebonden worden, of in combinatie met thermoforming. Afbeelding 2b toont deze tool.

Deze thesis heeft 4 contributies:

1. De DIY ontwerp en design principes achter de modules, inclusief de flexibele form, elektronica en connectors.
2. Een nieuwe DIY fabricatie techniek om flexibele PCB's te maken door middel van een combinatie van een CO2 en fiber laser.
3. Een software tool, inclusief communicatie protocols, om de topologie van verbonden modules te traceren en visualiseren, en zowel standaard plug-en play gedrag als gedrag gespecifieerd door de eind-gebruiker toelaat.
4. Een aantal voorbeelden van designs en use cases die de nieuwe mogelijkheden en het gebruik van SoftMod om te prototypen van nieuwe types van interactieve systemen tonen. Hierbij hoort ook de software tool en techniek die gebruikt wordt om een SoftMod schakeling om te zetten naar een high-fidelity prototype.

## SoftMod Concept en Use Cases

Deze sectie legt het algemene concept van SoftMod uit aan de hand van een walkthrough over het samenstellen van een SoftMod schakeling. Deze sectie geeft ook een overzicht van alle huidige ondersteunde modules, een beschrijving van het gedrag en een aantal use cases.

### Walkthrough

Deze walkthrough illustreert de werkwijze van het maken van een flexibele interactieve armband die als een loop-timer dient. Het proces begint door een SoftMod master-module, LED-module, knop-module en batterij-module aan elkaar te klikken, zoals

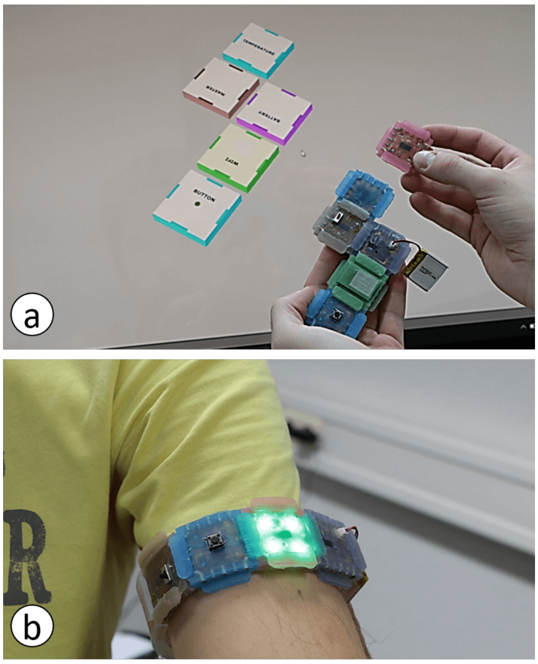



FIGURE 3: (a) Klik SoftMod modules aan elkaar om (b) een interactieve armband te prototypen.

getoond in figuur 3a. SoftMod herkent de topologie en past het standaard gedrag toe: als op de knop geduwd wordt zullen de leds gaan branden. Om dit gedrag te veranderen moet de master module aan een computer met de SoftMod IDE verbonden worden. Deze software visualiseert de topologie en laat toe het gedrag van alle modules aan te passen door gebruik te maken van IF-This-Then-That (IFTTT) regels of een scratch-achtige programmeer omgeving. In dit voorbeeld zal de knop wisselen tussen drie verschillende modi: 3 minuten lopen - 1 minuut wandelen, 5 minuten lopen - 5 minuten wandelen, en 10 minuten lopen - 5 minuten wandelen. De LEDs zullen altijd aan zijn als er gewandeld moet worden, en knipperen als er gelopen moet worden.

Modules kunnen makkelijk toegevoegd, verwijderd of gewisseld worden tijdens het prototypen. In dit voorbeeld worden verschillende lengtes van kabels gebruikt om een armband te creëren die past rond de arm (figuur 3b).

### Ondersteunde Modules

SoftMod modules kunnen steeds uitgebreid worden naargelang de vereiste functionaliteit. Momenteel zijn volgende types ondersteund: een (a) master-module (bruin) en zes types van slaaf-modules. Deze zijn: (b) input-modules (blauw), inclusief een temperatuur-module, knop-module, en accelerometer-module; (c) output modules (rood), met een standaard LED module en RGB LED module; (d) draadloze communicatie modules (groen); (e) batterij-modules (paars); (f) twee kabel-modules (transparant-wit); en (g) twee adapter-modules (transparant-wit). Elke schakeling moet exact één master bevatten, maar er kunnen tot 124 slaaf modules aangestuurd worden in één schakeling.

### Gedrag

SoftMod heeft twee technieken om het gedrag van een schakeling te bepalen. Dit gedrag kan eenvoudig gewisseld worden door de gebruiker aan de hand van een switch

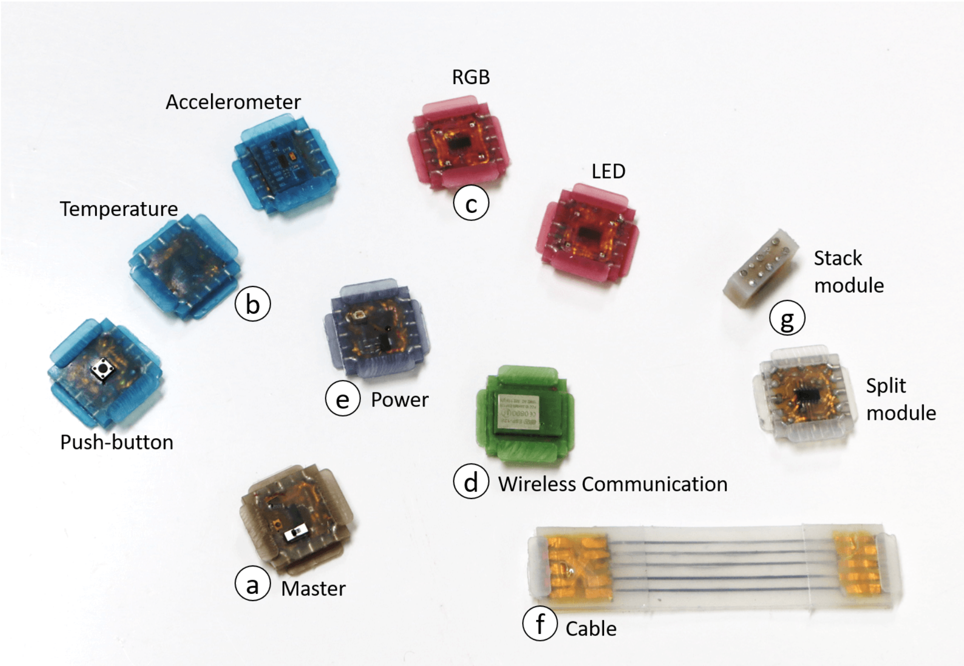



FIGURE 4: SoftMod ondersteunt zeven types van modules: (a) een master-module, (b) input-modules, (c) output-modules, (d) draadloze communicatie modules, (e) batterij-modules, (f) kabel-modules, (g) adapter-modules.

op de master-module.

- **Standaard gedrag**: In deze mode zal de master het gedrag van de modules bepalen aan de hand van de topologie van modules. Deze mode zorgt voor de plug-en-play aard van SoftMod omdat alle verbonden modules al meteen functionaliteit hebben zonder ze te programmeren.
- **Gebruiker-bepaald gedrag**: De master-module kan verbonden worden met een computer door een extra programmeer bord met USB interface dat zich verbindt met de magnetische connectors (figuur 5). De SoftMod IDE laat toe om het gedrag van een SoftMod schakeling aan te passen door ofwel gebruik te maken van standaard IF-This-Then-That (IFTTT) regels, ofwel door een S4A scratch-achtige programmeer omgeving te gebruiken[1], ofwel door het schrijven van C-code in de Arduino IDE.

## Use cases

Aangezien de vorm van SoftMod modules makkelijk te wijzigen is, heeft SoftMod een grote verscheidenheid aan use cases. In de eerste plaats kan SoftMod gebruikt worden als experimenteer toolkit voor kinderen en volwassenen. In de tweede plaats kan SoftMod gebruikt worden in wearables, zoals een interactieve armband (figuur 6a). SoftMod schakelingen zouden ook bestaande objecten kunnen verrijken door extra functionaliteit te voorzien (figuur 6b), of door embedded te worden in een object (figuur 6c). Tot slot kan SoftMod ook gebruikt worden als interactieve tattoos en kleding (figuur 6d).

[1] http://s4a.cat

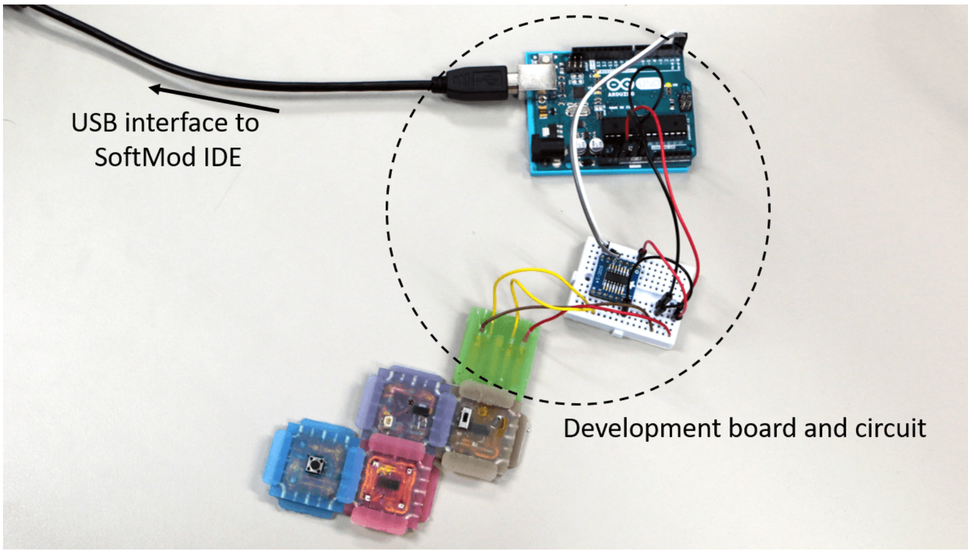



FIGURE 5: De master-module is verbonden via een extra programmeer bord met een computer die de SoftMod IDE draait.

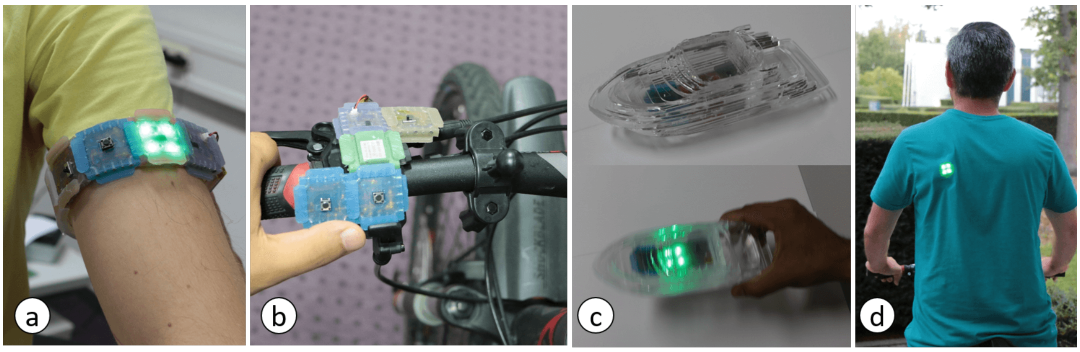



FIGURE 6: SoftMod ondersteund verschillende use cases: (a) een interactieve armband, (b) een fietsstuur uitgerust met knoppen voor pinkers, (c) een schakeling embed in een voorwerp en (d) leds gecast in een T-shirt om richting aanwijzingen te kunnen geven.

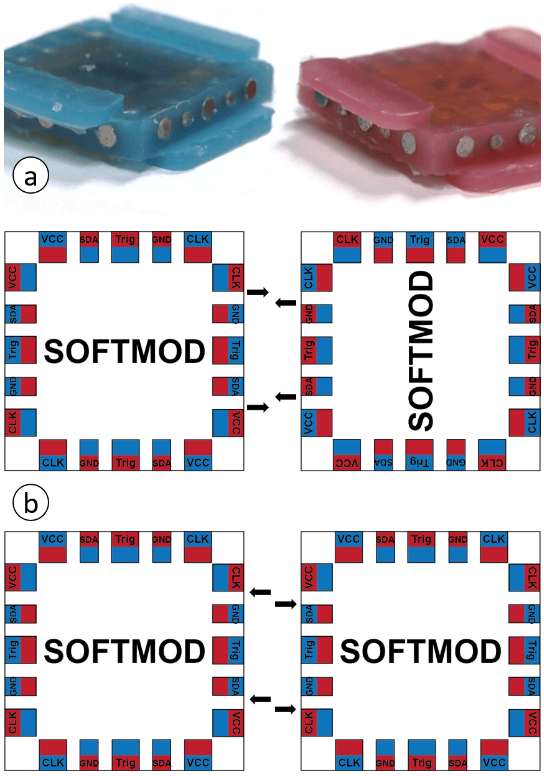



FIGURE 7: (a) SoftMod's connectoren bevatten magneten de (b) die zorgen dat de connectors ofwel aangetrokken ofwel afgestoten worden tot elkaar.

## Fabricatie: Modules en Connectoren

Deze sectie beschrijft het design van de modules en connectoren.

### Connectoren

Om een plug-en-play gedrag te ondersteunen maakt SoftMod gebruik van magnetische connectoren. Deze connectoren bevatten vijf neodymium cilinderachtige magneten, die ook als conductors gebruikt worden. Afbeelding 7 toont deze connectoren. Er is ook een test uitgevoerd om de sterkte van de connectoren te testen. Deze test wordt weergegeven in afbeelding 8. Twee connectoren kunnen tot 90 graden gebogen worden zonder elektrische conductiviteit te verliezen. De kabels, gemaakt met silicone shore 5, 5 mm dik aan de uiteindes en 3 mm dik in het midden, kunnen uitgetrokken worden tot 27% alvorens de magnetische connector los laat.

### Modules

Alle SoftMod modules bevatten een flexibele PCB, beschreven in volgende sectie. Deze PCB wordt gecast in silicone shore 5 met behulp van een laser-cut mold, gebaseerd op StackMold [69]. Om de magneten makkelijk te solderen, en om verlies van polariteit door de warmte van de soldeerbout tegen te gaan, is er ook een houder gemaakt die de magneten op hun plaats houdt. Modules hebben standaard een afmeting van 30 op 30 op 5 mm.

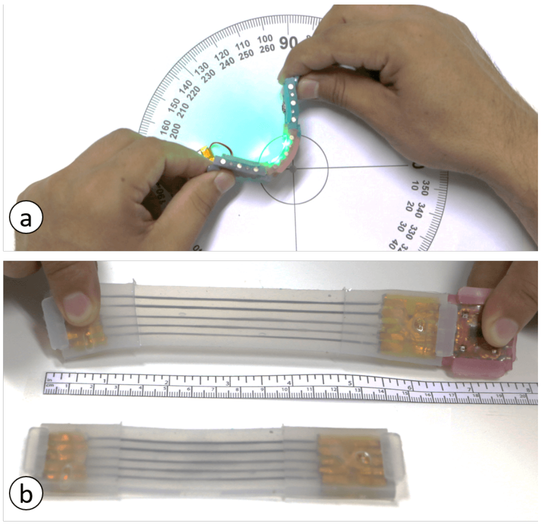



FIGURE 8: De magnetische connectoren zijn sterk genoeg om elektrische geleidend te zijn terwijl ze (a) gebogen en (b) uitgetrokken worden.

## Fabricatie: Flexibele PCB's

Om de flexibele PCB's te produceren is er een nieuwe techniek ontwikkeld die gebruik maakt van op elkaar gestapelde lagen van koper en Kapton plakband. Door gebruik te maken van een CO2 laser-cutter kan Kapton gesneden worden zonder het koper aan te tasten. Ook in omgekeerde richting kan een fiber laser koper snijden zonder de Kapton te snijden, al kan de Kapton wel verbranden. Hierdoor is het noodzakelijk een fiber laser calibratie te doen. Deze techniek is ook beschreven in een poster ingestuurd voor UIST 2019 ([39]).

## Architectuur: Elektrische Componenten, Protocollen en Algoritmes

### Elektronische componenten

Alle SoftMod modules bevatten dezelfde microcontroller, namelijk een ATtiny1614. Deze microcontroller is groot genoeg voor de meeste taken, heeft hardware ondersteuning voor de meeste communicatie protocollen en voorziet genoeg digitale in- en output pinnen. Slaaf-modules bevatten ook nog extra componenten afhankelijk van de functionaliteit van de module.

### Protocollen

Als inter-module communicatieprotocol wordt er gebruik gemaakt van I2C. Dit is een snelle bus-communicatie die enkel twee kabels nodig heeft. Slaaf-modules kunnen enkel door de master aangesproken worden door middel van een 7-bit adressering systeem. Om informatie van de master module naar de computer over te brengen wordt er gebruik gemaakt van een UART verbinding.

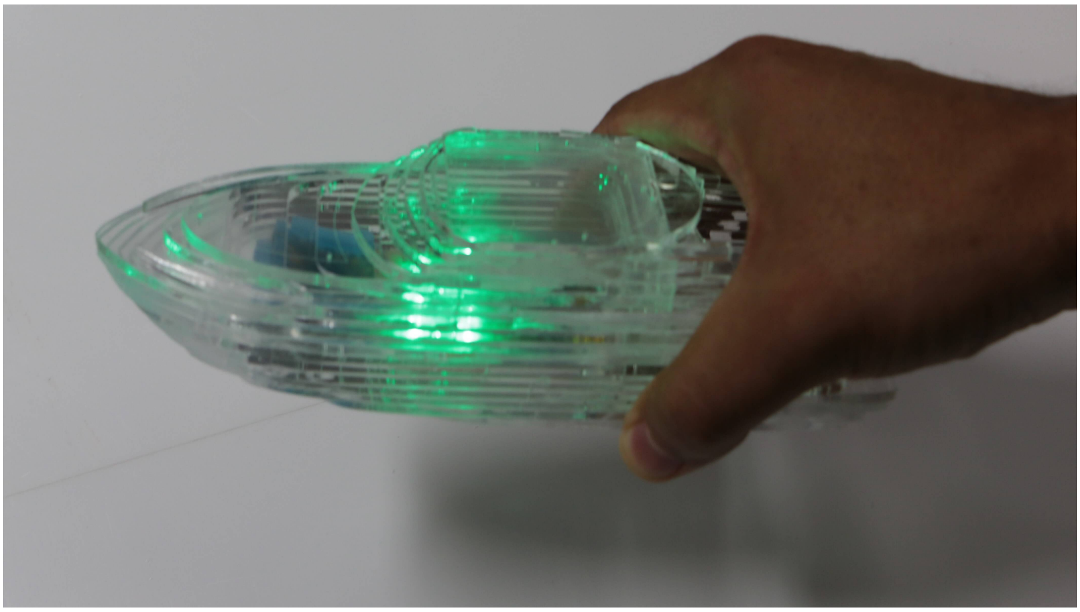


FIGURE 9: Een SoftMod schakeling ingebed in een boot gemaakt van plexiglas. De led gaat aan of uit als er met de boot geschud wordt.

### Algoritmes

SoftMod bevat enkele belangrijke algoritmes. In de eerste plaats is door het gebruik van I2C het aantal te verbinden modules beperkt tot 127. Indien elke slaaf-module standaard een uniek adres zou hebben, zijn er niet genoeg unieke adressen waardoor er bus-conflicten optreden door duplicate adressen. SoftMod heeft daarom een dynamisch adressering systeem dat unieke addressen uitdeelt aan de hand van een selectieprocedure gebaseerd op unieke 64-bit addressen en de detectie van bus-collisions.

Om de topologie te detecteren maakt SoftMod gebruik van een vijfde connector pin: de trigger. Deze pinnen voorzien point-to-point verbindingen met de naburig gelegen modules en kunnen data verzenden naar elkaar. Wanneer een nieuwe module aan de schakeling wordt toegevoegd, zal de master-module deze de opdracht geven om zijn buren te detecteren. De nieuwe slaaf-module doet dit door om beurt een interrupt te genereren op één van zijn trigger pinnen. Indien een module deze interrupt binnenkrijgt stuurt deze zijn I2C adres over de trigger pin naar de slaaf-module.

## SoftMod schakeling naar High-Fidelity Prototype

Er zijn vele technieken mogelijk om een SoftMod schakeling om te zetten naar een high-fidelity prototype. Zo kan een schakeling bijvoorbeeld gecast worden in resin of silicone om deze te versterken. Een schakeling kan ook volledig ingebed worden, getoond in figuur 9. Een SoftMod schakeling kan ook gebruikt worden om extra functionaliteiten toe te voegen aan een 3D object. Een andere mogelijke techniek is om een volledig nieuw circuit te genereren dat dezelfde karakteristieken heeft als de schakeling. Dit circuit zou zelfs geoptimaliseerd kunnen worden door één grote microcontroller te gebruiken die dezelfde functionaliteit als de schakeling bied.

SoftMod stelt een techniek voor die een bepaalde 3D positie op een object kan converteren naar een 2D positie. Met andere woorden laat deze techniek toe om virtueel componenten op een object te plaatsen waarna een schematische voorstelling van de

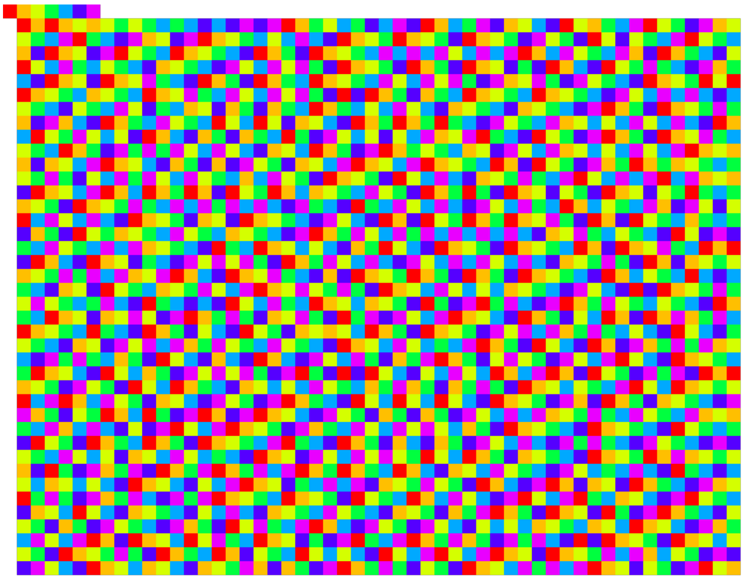


FIGURE 10: Het calibratie rooster dat gebruikt wordt voor de conversie van 3D punten naar 2D.

te creëren 2D schakeling getoond wordt. Deze techniek is geïmplementeerd in een software applicatie (zie afbeelding 2b) die gebruik maakt van een speciaal calibratie rooster, getoond in afbeelding 10, dat geplaatst moet worden rond het object. In dit rooster is elke 3x3 combinatie van kleuren uniek en het rooster kan zelfs uitgetrokken worden zonder problemen. Hierdoor kan er ook gedetecteerd worden welke transformaties het object heeft ondergaan, zoals bijvoorbeeld tijdens thermoforming. Met behulp van een het ingebouwde algoritme kan de conversie van 3D punten naar 2D gebeuren, waarbij er rekening gehouden wordt met de transformaties die op het object zijn uitgevoerd.

## Limitaties, Future Work and Conclusie

Deze sectie geeft een samenvatting van de limitaties en toekomstig werk van SoftMod, gevolgd door de conclusie.

### Limitaties en Future Work

In de eerste plaats heeft SoftMod een aantal technische limitaties. Ten eerste kan SoftMod uitgebreid worden met meer geavanceerde logica. De SoftMod IDE bevat momenteel nog geen ingebouwde programmeer omgeving. Ten tweede kunnen er altijd meer SoftMod modules gemaakt worden. Ook kan in een toekomstig connector design rekening gehouden worden met het vermijden van kortsluiting. Tot slot is de ATtiny1614 maar juist groot genoeg voor de code van de master-module. Een upgrade naar bijvoorbeeld een Microchip SAM L10[10] zou meer mogelijkheden bieden.

Naast de technische limitaties zijn er ook limitaties in de voorgestelde nieuwe manier van prototypen. Zo kunnen SoftMod modules bijvoorbeeld versterkt worden, maar zonder extra steun zullen deze hun vorm niet kunnen blijven behouden aangezien ze gemaakt zijn van silicone. Daarnaast heeft de voorgestelde "SoftMod schakeling naar high-fidelity prototype" techniek een aantal limitaties. In de eerste plaats moet het mogelijk zijn om het calibratie patroon aan te brengen op het object. Ten tweede is het soms makkelijker SoftMod modules rechtstreeks te plaatsen op een object. In een toekomstige versie zou de software automatisch, vertrekkende vanuit een gebouwde schakeling, een volledig nieuwe circuit, inclusief bijhorende molds en houders, kunnen genereren.

## Conclusie

In deze thesis hebben we SoftMod, een modulaire elektronica kit bestaande uit flexibele modules die in elkaar klikken om nieuwe interactieve systemen te prototypen, gepresenteerd. SoftMod heeft kwaliteiten voor zowel kinderen als volwassenen als het aankomt op eenvoudig plug-en-play gedrag, alsook geavanceerd gebruikers-bepaald gedrag. In deze thesis hebben we ons software framework, elektronica en mechanische connector designs, en prototyping procedures om SoftMod modules te maken uitgelegd. We hopen dat onze toolkit designers, onderzoekers en artiesten aanzet om nieuwe ideeën voor elektronische systemen, zoals wearables, interactieve huid en textiel te onderzoeken. Aangezien het proces om SoftMod modules te maken volledig DIY is, hopen we ook dat de community verder bouwt op onze ideeën.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Electronic development boards, such as Arduino [5], Raspberry Pi[66], and Phidgets [26], support hobbyists in making electronic systems. However, these boards require basic knowledge of electronics and programming. To further lower the barrier for prototyping with electronics, plug-and-play electronic modules allow for making sensor systems by simply connecting modules. Popular examples include LittleBits [13] that snap together and MakerWear modules [36] that interconnect through a flexible carrier. While these modules make prototyping electronic behavior convenient, the shape of the resulting electronic systems is mostly dictated by the functionality as reconfiguring modules in different shapes results in different behavior. These modular kits are therefore often used for play and curiosity-driven learning.

To support maker enthusiasts in prototyping electronic systems with more intricate form-factors and sensors embedded in materials, researchers explored techniques to lower the barrier for crafting with electronics on paper [57, 59], textiles [29], and silicones [51] through workshops [47], online tutorials [56], and software tools [59]. Although these approaches result in electronic systems that are more permanent (sturdy) and closely resemble the look-and-feel of a final product, the workflows are often tedious which significantly increases the time required for making design variations.

This thesis presents a new prototyping process for non-engineers. In the first place SoftMod is developed. SoftMod (Soft Modules) is a soft modular electronic kit that empowers people without an engineering background to make electronic systems in a desired shape by snapping together soft, flexible modules, as shown in figure 1.1. SoftMod supports modules for sensing, output, processing, power, and wireless communication. Unlike existing plug-and-play modular kits, such as LittleBits [13] and MakerWear modules [36], the topology of a SoftMod assembly is tracked, and the default behavior is adjusted accordingly. As such, the topology of interconnected modules can fit the desired shape of the electronic system and does not influence the behavior. More advanced logic can be specified using the SoftMod IDE. SoftMod modules also interconnect through soft stretchable cables, which make it possible to realize larger 2D or even 3D structures. On the one hand, SoftMod empowers and stimulates designers and enthusiasts to explore ideas for new types of interactive systems, such as wearables and interactive skin and textiles. On the other hand, SoftMod also appeals to children and adults to play and thinker with sensors without risking any damage to components or one's safety. SoftMod is also published in a paper submitted for TEI 2020 ([38]).

Next, the fabrication process for SoftMod modules is fully DIY compatible and consists of extremely flexible PCBs, as shown in figure 1.2, produced by a novel DIY fabrication technique. This technique stacks layers of Kapton and copper tape on top of each other. The key to this approach is the combination of a CO2 and fiber laser. By

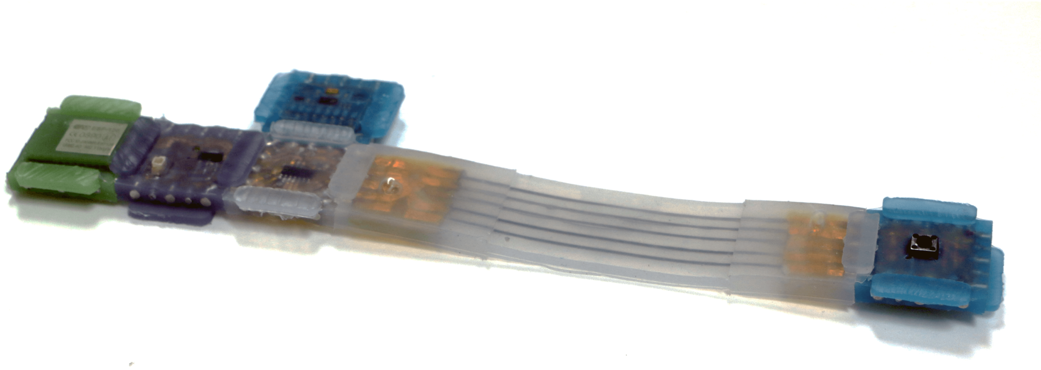


FIGURE 1.1: A SoftMod assembly.

using one of the lasers, we can selectively cut either copper or Kapton and also cure solder paste for making VIAs and for soldering components. As those circuits consist of Kapton and copper, they are highly conductive and thus support high-frequency signals, such as I2C. Also, magnetic connectors are precisely mounted and cast in place using laser-cut molds. Therefore, this thesis also provides details on the engineering and fabrication procedure for assembling SoftMod modules. The community can learn and adopt these approaches and join our effort in maintaining and extending the SoftMod kit.

Lastly, SoftMod assemblies can be changed from quick and easy to change prototypes to high-fidelity prototypes by casting them into objects or reinforcing them. This aspect enables evolutionary prototyping since SoftMod assemblies can easily be changed into a new iteration of the prototype. This thesis presents a software tool that visualizes SoftMod modules on 3D objects to map out the required 2D shape of the circuit. Figure 1.3 shows this tool. This process is useful for when users, for example, want to wrap their assemblies around specific objects where the location of modules is essential or when assemblies are used in combination with thermoforming.

This thesis has four main contributions:

1. The DIY engineering and design principles behind the modules, including the soft form factor, the electronic design, and mechanical interconnections. (Chapter 4)

2. A novel DIY fabrication technique for making flexible PCBs by using the combination of a CO2 and fiber laser. (Chapter 5)

3. An embedded software framework, including communication protocols, to track the topology of interconnected modules and allow for basic plug-and-play behavior as well as end-user specified behavior. (Chapter 6)

4. A set of example designs and use cases that showcase the novel possibilities and the utility of SoftMod for prototyping novel types of interactive systems. This also includes a software tool and technique to convert a SoftMod assembly to a high-fidelity prototype. (Chapter 7)

This thesis is structures as follows. Chapter 2 starts by giving an overview of all relevant related work. Next, chapter 3 describes the general concept of SoftMod while also giving the most fundamental use cases. Chapter 4 continues the thesis by

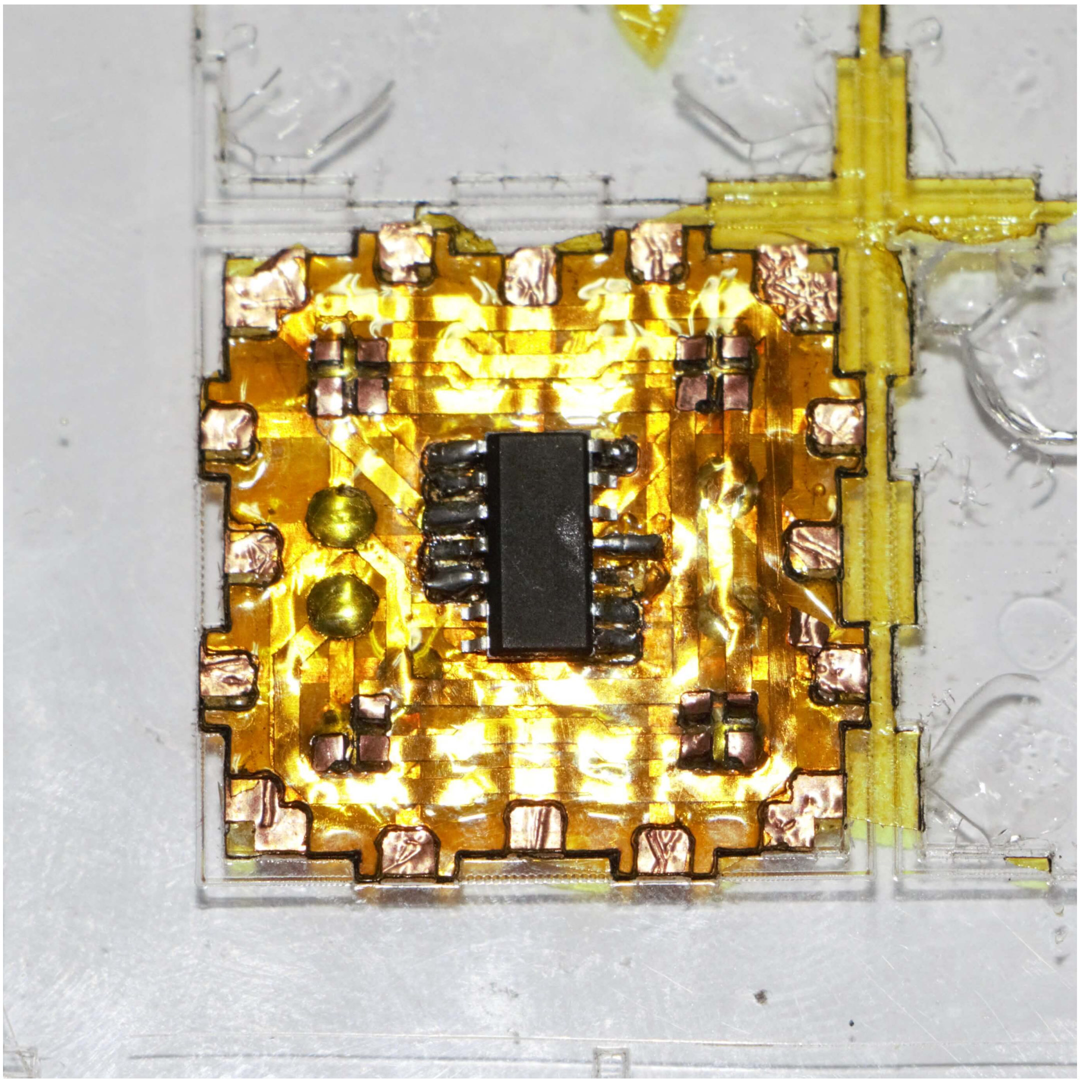


FIGURE 1.2: A custom created flexible PCB.


FIGURE 1.3: Software tool used to convert a SoftMod assembly placed on a 3D shape to a 2D assembly.

explaining the fabrication of the soft modules and connectors, followed by chapter 5, which contains the manufacturing process of the flexible PCB's. Chapter 7 shows the process of creating high fidelity prototypes from a SoftMod assembly. Finally, chapter 8 discusses the limitations of SoftMod and also gives an overview of future work. This chapter also concludes the thesis.

# Chapter 2

# Related Work

This chapter discusses and compares similar toolkits and electronic systems with each other to provide an overview of the current domain. SoftMod builds upon and relates to tools for lowering the barrier to prototype electronic systems and Do-It-Yourself (DIY) crafting of electronic systems. This chapter is divided in three big parts: the first section 2.1 compares similar tools for prototyping electronic systems. Next, section 2.2 explains the programming aspect of those systems. Finally, section 2.3 dives deeper in the crafting aspect of electronic systems.

## 2.1 Tools for Prototyping Electronic Systems

Electronic development boards, such as Arduino [7] and Raspberry Pi [66] make it affordable and convenient for maker enthusiasts to prototype electronic systems. However, these toolkits still require to have a basic experience in electronics and programming. This knowledge is needed to correctly connect and program components without destroying them. To allow users with limited to no electrical expertise to build electronic systems, several alternatives exists, such as Phidgets [26], .Net Gadgeteer [32] (shown in figure 2.1), Lego Mindstorms [49], and MakeBlock [45]. These systems abstract more complex elements such as wiring of components and inter-modules communications. As such, they allow for prototyping with high-level electronic modules without requiring the user to handle low-level components, such as resistors, capacitors, shift-registers, or amplifiers.

Researchers also experimented with embedding a communication system in layered materials, instead of using regular cables. In these approaches, conductive and insulating rigid [37], flexible [70], or textile [25] materials are stacked. Figure 2.2 shows an example of VoodooIO [70]. By special designed connector pins modules make contact with the correct corresponding layers. Such approaches avoid having many cables, but still allow components to be configured in various layouts.

Another approach of connecting modules together is by using wireless communication protocols, such as WiFi or Bluetooth. Those modules act as standalone modules and can be connected to everything that implements the same protocol. They are often used in IoT solutions like Particle [55] and Sonoff [63]. Also the Calder toolkit [40] makes use of wireless links. Examples of such a modules are the ESP8266 [22] or Microchip's SAMW25 [74], which is advertised as a standalone IoT solution with both a microcontroller and WiFi SoC in one complete module package. Connecting and programming those modules, however, requires much experience.

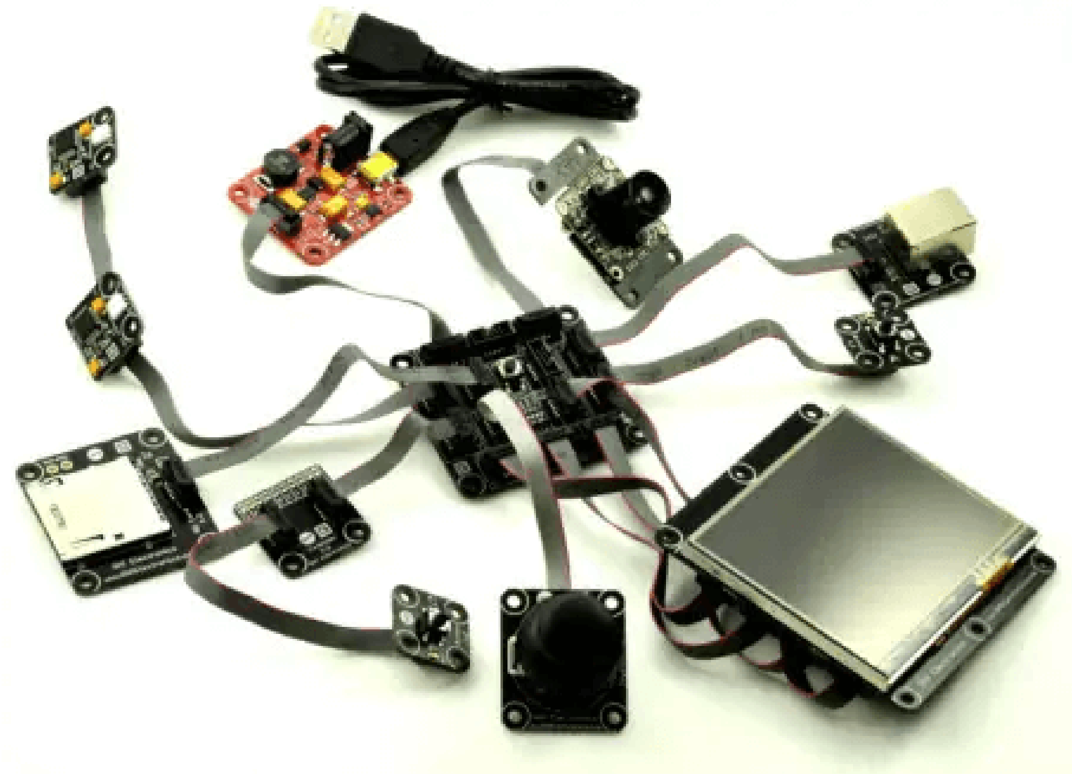


Figure 2.1: .Net Gadgeteer [32] modules, interconnected using ribbon cables.

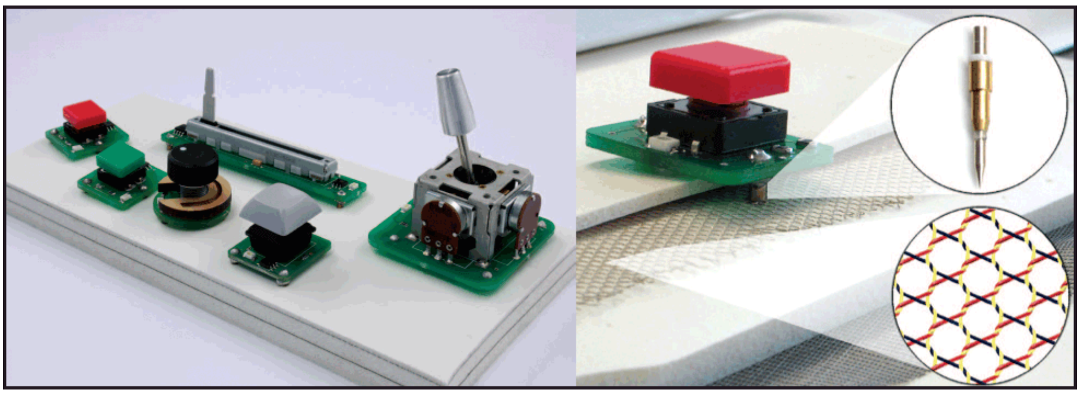


Figure 2.2: VoodooIO [70] modules interconnected by placing them in layered conductive materials.

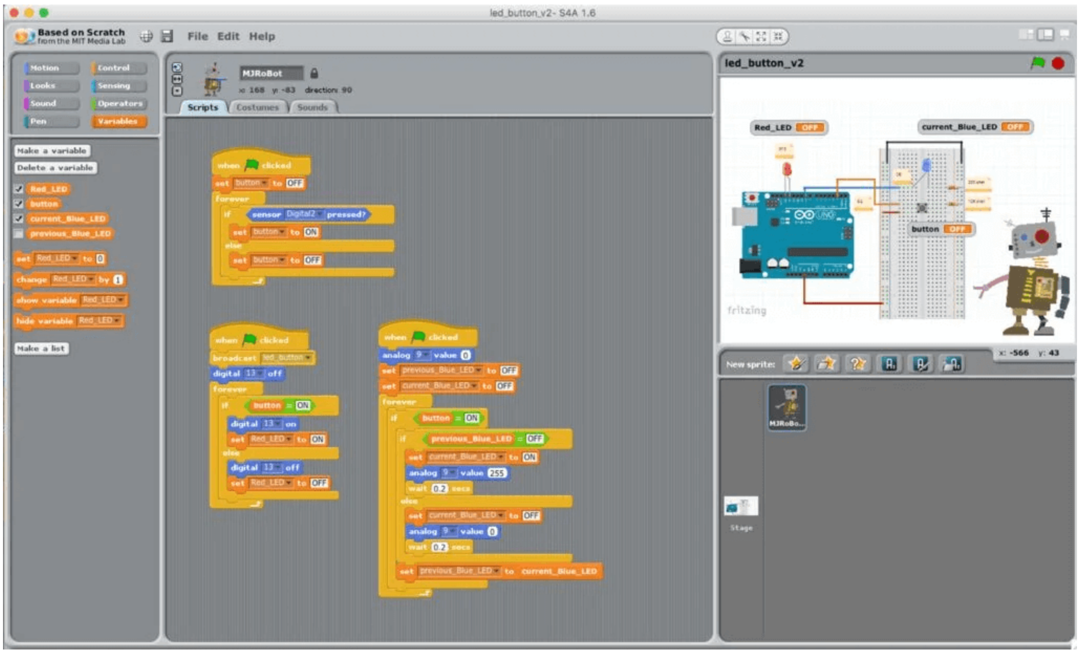



FIGURE 2.3: Scratch for Arduino [60] to program an Arduino by using Scratch.

## 2.2 Tools for Programming Electronic Systems

Besides electronics, researchers also aim to lower the barrier for programming electronic systems using IFTTT [4, 34], state-charts like d.tools [30], and scratch-like approaches [19, 48, 60] (shown in figure 2.3). IFTTT (If-This-Than-That) is a set of simple rules that are executed if a specific condition is true. D.tools is a toolkit that embodies an iterative-design-centered approach to prototyping information appliances.

To empower true novices and children to build electronic systems, several toolkits for sensor systems do not require programming as they assign a basic behavior to every module [13, 15, 36, 62]. By composing these building blocks, various types of behavior can be realized. For example, when a button connects to an RGB LED, this button can control the state of the LED. However, when a temperature sensor replaces the button, the color of the LED shows an indication of the current temperature. Popular examples include LittleBits [13], for rigid sensor systems as shown in figure 2.4, MakeWear [36] for prototyping on textiles, and roBlock [62] for robotic systems. LightUp[17] is similar to LittleBits as both use blocks that magnetically connect together, but LightUp provides an "informational lens" which displays the behaviour of the system by augmenting a photographed circuit. Although these plug-and-play kits make it very convenient for making sensor systems, the desired sensor behavior dictates the shape and thus the form-factor.

In contrast to these approaches, SoftMod empowers non-engineers to prototype desired 2D and 3D shapes using a soft plug-and-play kit. By tracking the topology of modules in real-time, a basic sensor behavior is assigned. At the same time, more advanced users can alter this basic sensor behavior using visual programming paradigms without requiring changes to the form-factor of a SoftMod assembly.


FIGURE 2.4: LittleBits [13] modules have a predetermined behaviour.

## 2.3 Crafting Electronic Systems

Embedding off-the-shelf electronic components in substances often result in circuits that are seamlessly embedded in artifacts. Examples include the Arduino Lilypad [14] and Adafruit Flora [64] that embed electronic circuits in fabrics by stitching them with conductive threads. These approaches, however, require crafting skills that can be acquired through workshops [47, 58] and online tutorials [56]. Perner Wilson et al. [56] argued that unlike construction kits, which are too general-purpose, crafting approaches encourage personal expression and learning.

Various crafting approaches have been presented to embed electronic circuits in materials. Examples include the e-TAGs [41] that embed ribbon cable connectors in textiles to facilitate attaching and swapping electronic components. Instead of using rigid cables and connectors, i*CATch [52] uses spring snap buttons. These buttons are interconnected using conductive strips that attach to fabrics using an iron-on adhesive. The TeeBoard [53] uses a similar technique but offers a breadboard-like layout, embedded in a shirt, as a universal electronic prototyping platform for textiles. To partially automate stitching of electronic systems in textiles, Sketch&Stitch [29], shown in figure 2.5, uses sketches of electronic circuit in combination with computer vision to drive a computerized embroidery machine.

In addition to textiles, several research projects also explore crafting techniques for embedding electronics in paper by interconnecting electronic sticker-like components [24, 31] using copper [58, 61] or conductive inks [59]. With more advanced crafting techniques, circuits can also be embedded on small-scale polymer films [44] or silicone [51, 72, 77]. However, despite these efforts to streamline crafting processes, these techniques often remain tedious and error-prone. In contrast, SoftMod is a soft modular platform that bridges the gap between crafting and construction kits. SoftMod allows for making a wide variety of 2D and 3D shapes, so they can be used to prototype

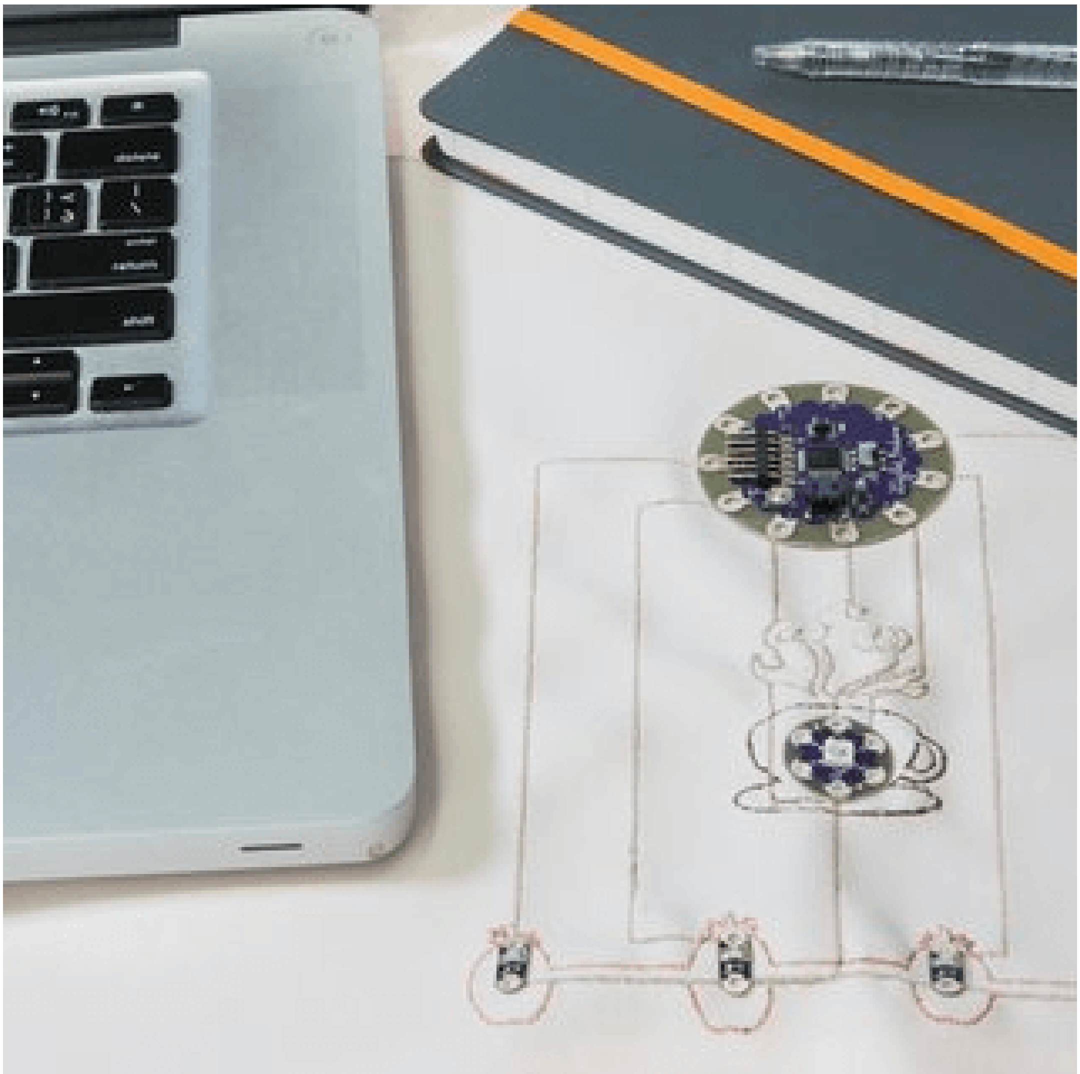


FIGURE 2.5: Sketch&Stitch [29] uses sketches of electronic circuit.

early-stage wearable and textile interfaces that traditionally would require advanced crafting techniques.

# Chapter 3

# SoftMod Concept and Use Cases

This chapter explains the general concept of SoftMod, together with use cases of SoftMod. The first section 3.1 starts by giving a walkthrough of the creation of a SoftMod assembly. Next, section 3.2 gives a high-level overview of all currently supported modules and their functionality, followed by section 3.3 that gives an overview of the general behavior of an assembly. Finally, section 3.4 will go over the use cases of SoftMod.

## 3.1 Walkthrough

The following walkthrough illustrates the process of making a soft interactive armband that serves as an interval timer while running. It consists of indicators notifying the user to switch between walking and running and a button to configure the interval. The final design is shown in figure 3.1b and has the same functionality and a similar look-and-feel as the interval timer prototyped in Silicone Devices [51], using state-of-the-art crafting techniques. With SoftMod, a similar prototype is realized in a fraction of the time (30min vs. a full working day).

The process starts with snapping together a SoftMod master module, an indicator module, and a push-button module. As shown in figure 3.1a, the embedded magnetic connectors make it very convenient to link modules. When powering the master module with a battery, SoftMod recognizes the topology of all connected modules and assigns a default behavior to the sensor system. In this configuration, all four LEDs of the LED module light up when touching the button. To be able to update the default behavior, the master module has to connect a desktop computer running the SoftMod IDE. This software environment visualizes the topology and allows for reconfiguring the behavior of all modules using IF-This-Then-That (IFTTT) rules or a scratch-like visual programming environment (figure 3.1a). In this example, the interval timer has four modes: 3min running - 1min walking, 5min running - 5min walking, and 10min running - 5min walking. The push-button is programmed to switch the mode, and the LEDs are configured to be powered while walking and blink when running.

Modules can easily be added, removed, or swapped during the prototyping stage. In this example, the user tests different cable modules to connect the first and last module and realizes an armband that matches the size of the arm (figure 3.1b).

## 3.2 Supported Modules

Every interconnected SoftMod assembly consists of a single master module and slave modules. While slave modules embed electronic functionalities, the master module controls the behavior of the slave modules and can be programmed. All modules have the same dimensions of 30x30x5mm. As shown in figure 3.1, the current version

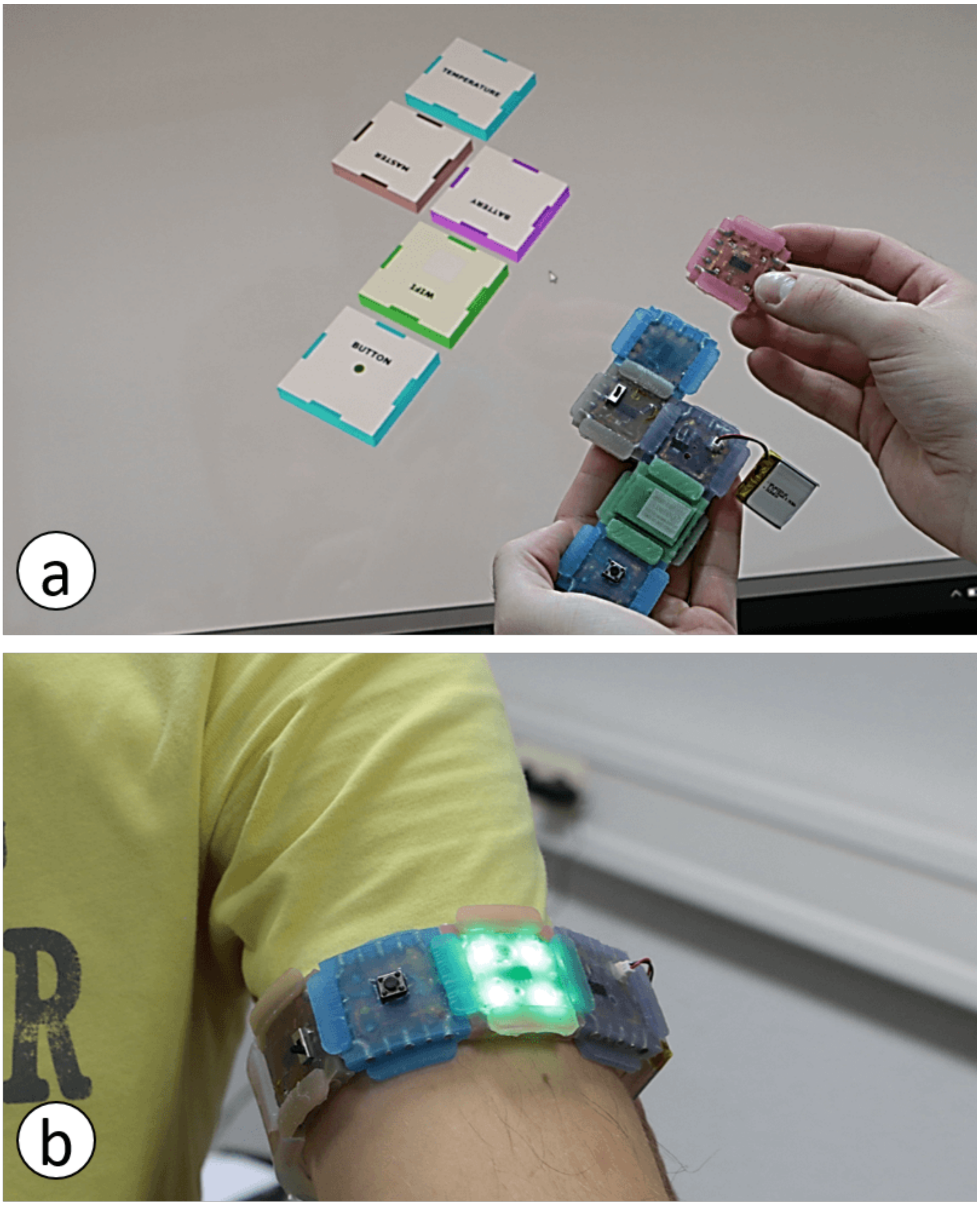



FIGURE 3.1: (a) Snapping together SoftMod modules to (b) prototype an interactive armband.

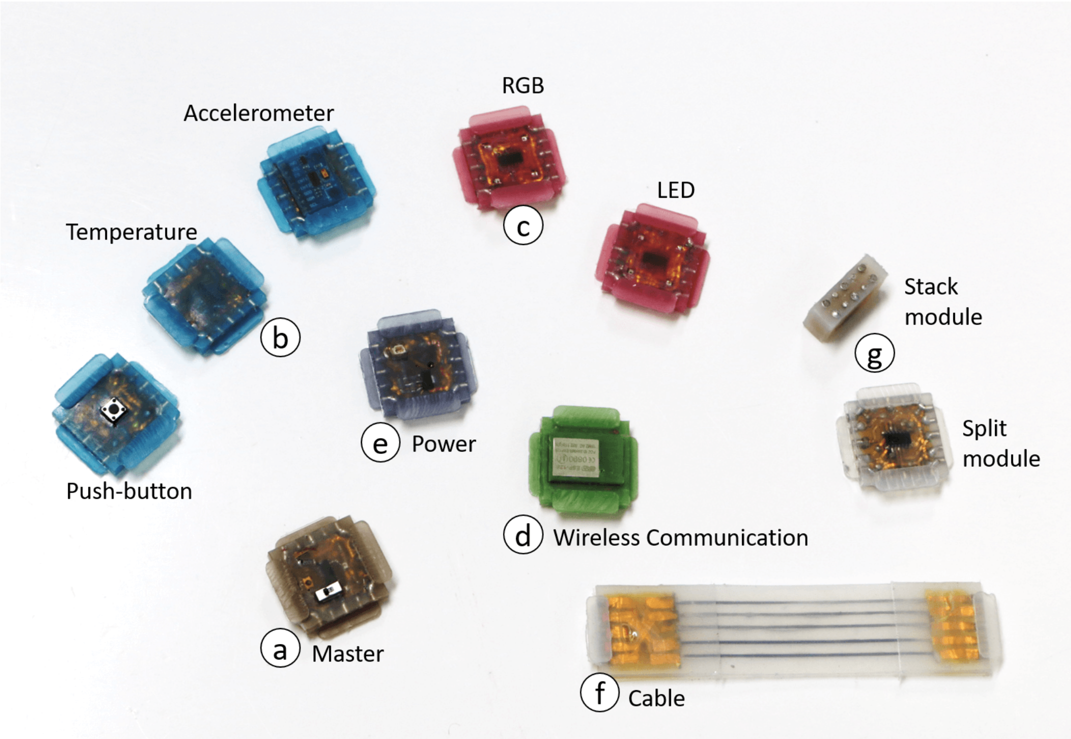



FIGURE 3.2: SoftMod supports seven types of modules: (a) a master module, (b) input modules, (c) output modules, (d) wireless communication modules, (e) power modules, (f) cable modules, (g) adaptor modules.

of the SoftMod system consist of (a) a master module (brown), and six types of slave modules: (b) input modules (blue), including a temperature module, push-button module, and accelerometer module; (c) output modules (red), including a standard LED module and RGB module; (d) wireless communication modules (green); (e) power modules (purple); (f) two cable modules (transparent-white); and (g) two adaptor modules (transparent-white). While exactly one master module needs to be present in any SoftMod assembly, up to 124 slave modules can be controlled by a single master module.

All modules, except the cable modules, embed four magnetic connectors to realize flexible 2D shapes. Cable modules shown in figure 3.2f, embed two magnetic connectors. They do not add functionality to the assembly but allow for configuring modules in intricate shapes beyond 2D grid layouts. SoftMod supports cable modules of different lengths. Each cable module embeds two magnetic connectors. In contrast to other types of slave modules, cable modules are in addition to being soft and flexible fully stretchable, as they consist of a liquid conductor (Section 4.2). SoftMod also comes with two adaptor modules that allow for (1) stacking of modules (figure 3.3a) and (2) branching a single magnetic connector into three connectors (figure 3.3b). The latter adaptor split module and has the same shape as the slave modules. Stacking of modules is especially useful for components that do not require direct visual attention or user interactions, such as the master module, power modules, accelerometer modules, and wireless communication modules. Stacking these modules oftentimes facilitates making specific shapes, such as interactive armbands that fit the arm.

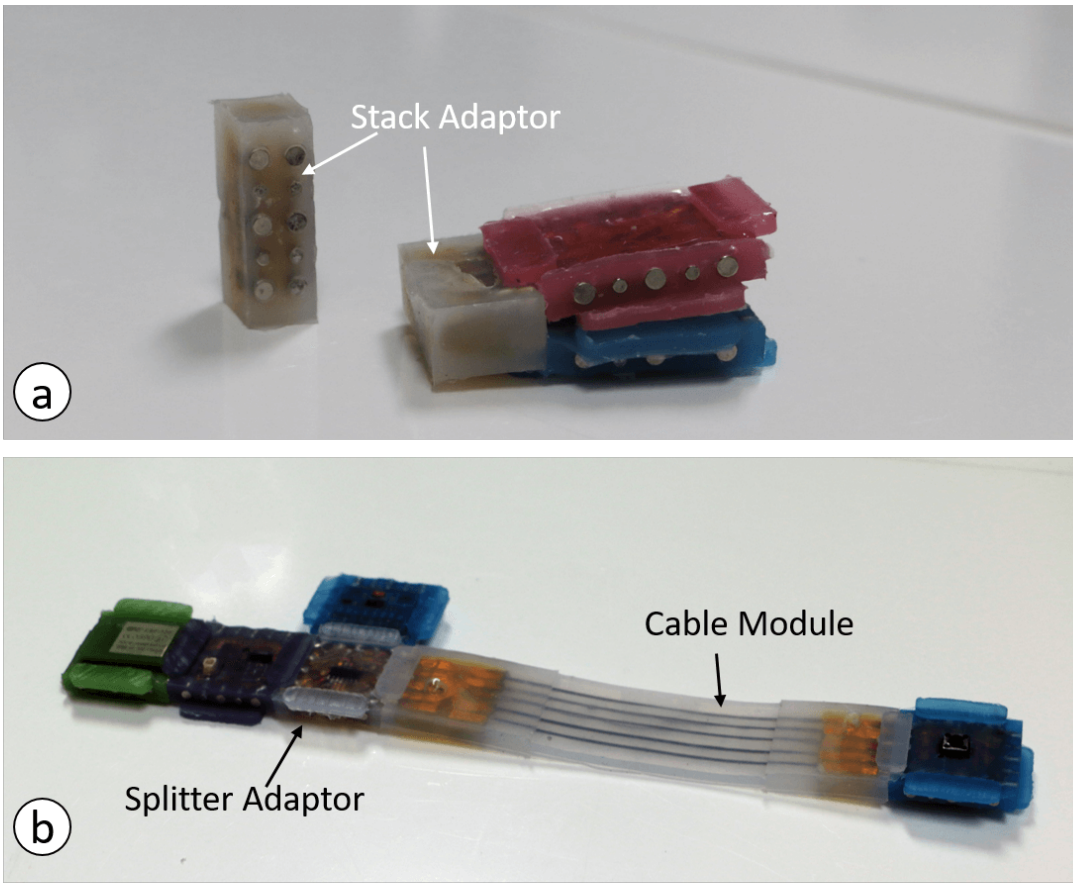



FIGURE 3.3: Two types of adaptor modules: (a) a stacking module, (b) a splitter module.

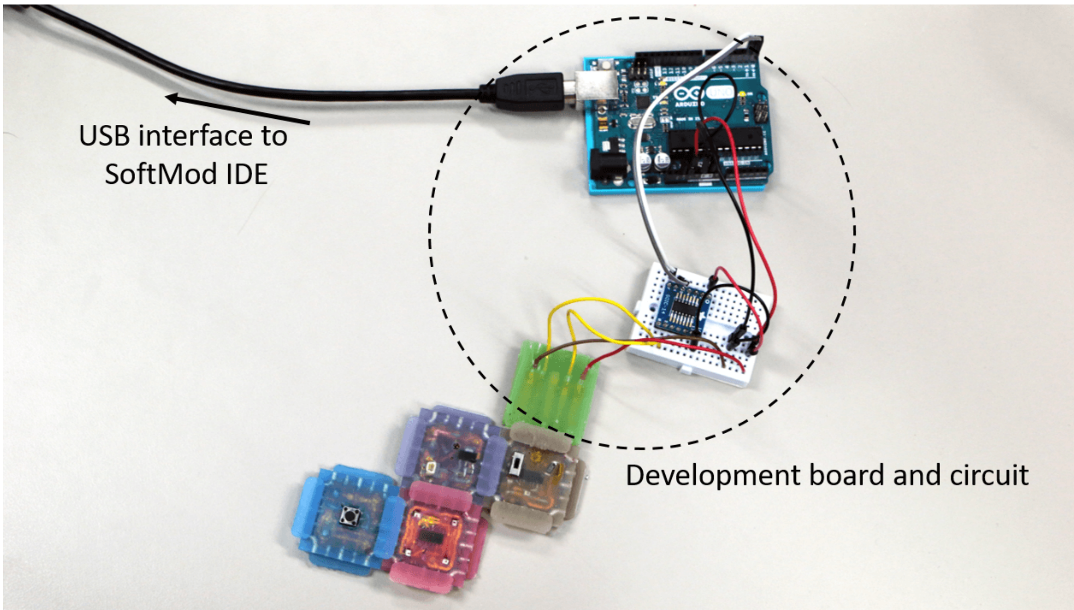



FIGURE 3.4: The master module is connected via an additional development board and circuit to a desktop computer running the SoftMod IDE.

## 3.3 Behaviour

SoftMod supports two techniques for defining the behavior of an assembly of modules:

- **Default behavior**: The master is configured to assign a default behavior to the slave modules based on their topology. This mode makes the best use of the plug-and-play nature of SoftMod as modules connected to the assembly immediately demonstrate some functionality. With current default behavior, an output module will turn on when controlling neighboring input modules. More advanced and intelligent default behaviors can be supported in the future.

- **User-specified behavior**: The master module is connected to a desktop computer through an additional development board and circuit that connects to the master module using a magnetic connector (figure 3.4). This additional development board offers a USB interface for serial communication with the desktop. The topology of the assembly is visualized in real-time in the SoftMod IDE. This software environment allows end-users to specify the behavior of a SoftMod assembly using basic IF-This-Then-That (IFTTT) rules, using the S4A scratch-like programming environment[1], or by writing C-code in the Arduino IDE. The user-specified code is then uploaded to the master module by pressing the program button on the master module. In a later iteration this button can be automated by a magnetic reed switch.

These two modes allow for using SoftMod both as a basic plug-and-play kit as well as a modular programmable electronic kit. Once user-specified behavior is loaded on a master module, the user can switch back to the default behavior switching the switch on the master module. After this switch, the master module needs to be reset in order to let the changes take effect.

[1] http://s4a.cat

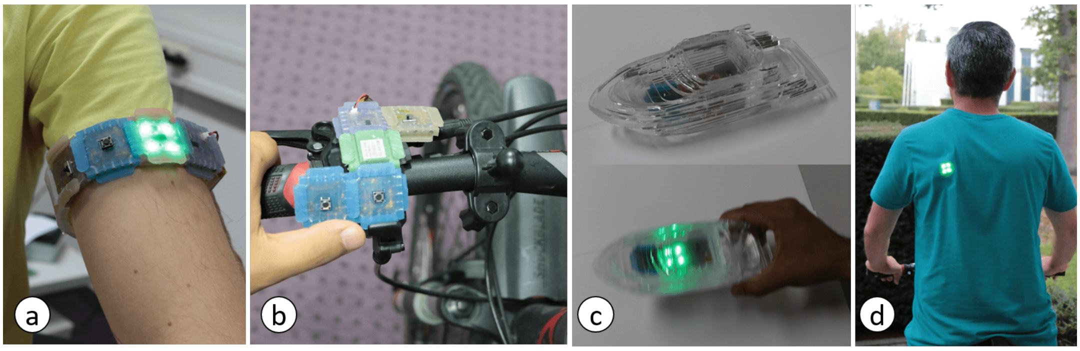



Figure 3.5: SoftMod allows for prototyping various novel types of interfaces: (a) an armband, (b) enriching the handle bar of a bike with blinker controls, (c) embedding interactivity in fabricated objects, (d) blinkers embedded in a shirt.

## 3.4 Use Cases

SoftMod assemblies are easy to change in shape using plug-and-play connectors. Additionally, the variety of programmable features allows for specifying the desired behavior. Therefore, SoftMod stimulates design explorations and allows users to materialize and test their ideas in small design iterations.

As the modules support making a wide variety of shapes, they can augment objects with interactivity. Some experiments were conducted in order to transition SoftMod assemblies to more permanent and robust electronic systems. By casting an additional layer of DIY casting silicone or dissolvable beeswax over the top of the modules, SoftMod assemblies can be reinforced temporarily or permanently. Such techniques allow for moving low-fidelity modular designs to high-fidelity electronic systems. Chapter 7 describes this technique more in-depth. The following sections discuss several use cases and example designs that SoftMod enables.

### 3.4.1 Curiosity, Play, and Experimentation

The plug-and-play magnetic connectors make it convenient to snap together modules and make a working electronic system without training. Therefore, it empowers children and adults to play and thinker with sensors without risking any damage to components or one's safety.

### 3.4.2 Wearables

SoftMod modules are flexible and allow for connecting modules in a loop. As shown in figure 3.5a, this enables prototyping of interactive bands that fit, for example, the chest, arm, or wrist. Such bands are comfortable to wear and take off using the magnetic connectors or the cable modules that allow for stretching over the body. When lots of modules are required, stacking modules can help to make small bands. In contrast to commercially available interactive bands, such as health monitoring wristbands, SoftMod wearables have a personalized functionality and shape and are easy to reconfigure, as demonstrated in the Walkthrough (Section 3.1).

### 3.4.3 Enriching Existing Objects

By attaching SoftMod assemblies to existing objects, they can alter, extend, or add interactivity to objects. Soft modules attach to objects with various curves using tape, fasteners, or Velcro. Figure 3.5b shows the handlebar of a bike augmented with two buttons that control blinkers on the biker's back (figure 3.5d). The blinking LEDs are also made with SoftMod and intercommunicate with the buttons using SoftMod's wireless communication module.

### 3.4.4 Making new Interactive Objects

As SoftMod supports making interactive 2D and 3D shapes, they can be embedded inside a fabricated object to add interactivity. Figure 3.5c shows an example of a laser-cut acrylic object embedding a SoftMod assembly. In this example, embedded LEDs light up when shacking the object. Alternatively, a SoftMod assembly can be inserted in a laser-cut mold structure that is then filled with casting material to realize more permanent interactive forms, similar to StackMold [69].

### 3.4.5 Interactive Skin and Textile

Soft modules also conform well to skin and textiles and are therefore suitable for prototyping interactive tattoos and clothes. Best results are achieved when casting a thin layer of DIY casting silicone on top of the modules to adhere the assembly to skin or textiles. Casting silicone over the modules also ensures the magnetic connectors are sealed and thus waterproof. Figure 3.5d shows an example in which an LED, wireless communication, and master module are embedded in a T-shirt to realize blinkers while riding the bike. Traditionally, prototyping interactive tattoos and textiles, require lots of materials, machinery, and DIY knowledge [29, 72]. SoftMod makes prototyping low-fidelity versions of these novel interfaces available to true novices.

# Chapter 4

# Fabrication: Modules and Connectors

This chapter contains the first part of the fabrication process and describes the design iterations of the fabrication of soft modules and connectors. SoftMod is the first soft modular electronic kit. Therefore standard manufacturing techniques and assembly lines for making rigid PCBs and connectors can not be used. Instead, state-of-the-art DIY fabrication procedures [39, 51, 69] are adopted and extended to create the soft modules, including the electronics design, flexible form factor, magnetic connectors, as well as stretchable cable modules. As a result, the SoftMod electronic kit can be replicated and extended by the DIY and maker community with machinery available in many FabLabs and MakerSpaces.

Section 4.1 starts this chapter by going over all different connector designs and fabrication techniques. Secondly, section 4.2 shows the process of creating different types of modules. This chapter will not go more in-depth on the fabrication of the flexible PCBs as this is described in chapter 5.

## 4.1 Fabrication of Connectors

Creating a suited connector for SoftMod is not a simple one-iteration process. Subsection 4.1.1 starts this section by describing the basic requirements for the connectors. Next, subsection 4.1.2 continues by explaining the most important iterations. Finally, subsection 4.1.3 goes deeper in on the molds used to create the connectors.

### 4.1.1 Connector Requirements

First of all, modules should easily snap together while not releasing too quick when being pulled. Secondly, the connector has to have five pins to support the implemented communication protocol, described in chapter 6. Connectors also have to be Poka-Yoke constrained to ensure modules only connect when correctly oriented. This constraint is key to the plug-and-play nature of the modules and avoids modules not functioning correctly or short-circuits. Poka-Yoke constraints, however, require a careful design mainly when connectors consist of soft silicone and thus can stretch.

### 4.1.2 Connectors

The following paragraphs describe the essential iterations that went over the connectors. The biggest obstacle is that all modules must be similar because each module should be able to connect with every other module. In a module, connectors laying on opposite sides are always the same.

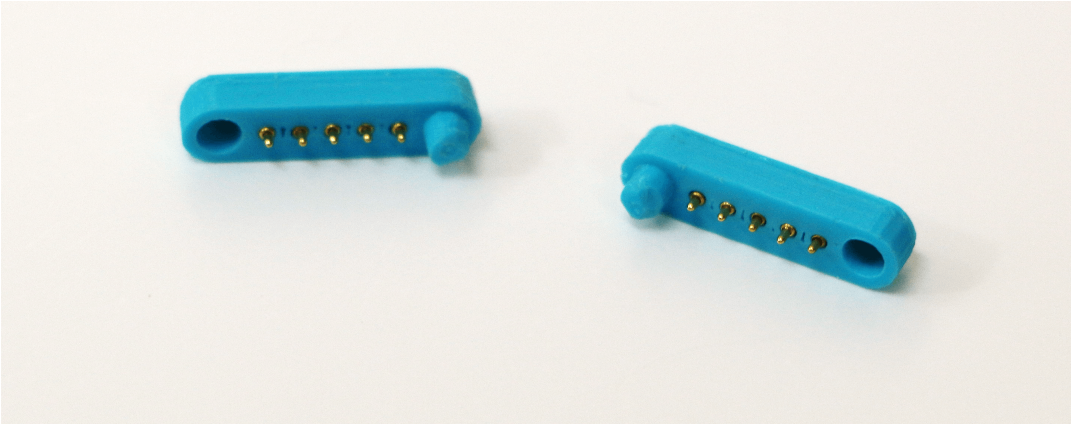


FIGURE 4.1: 3D-printed connector.

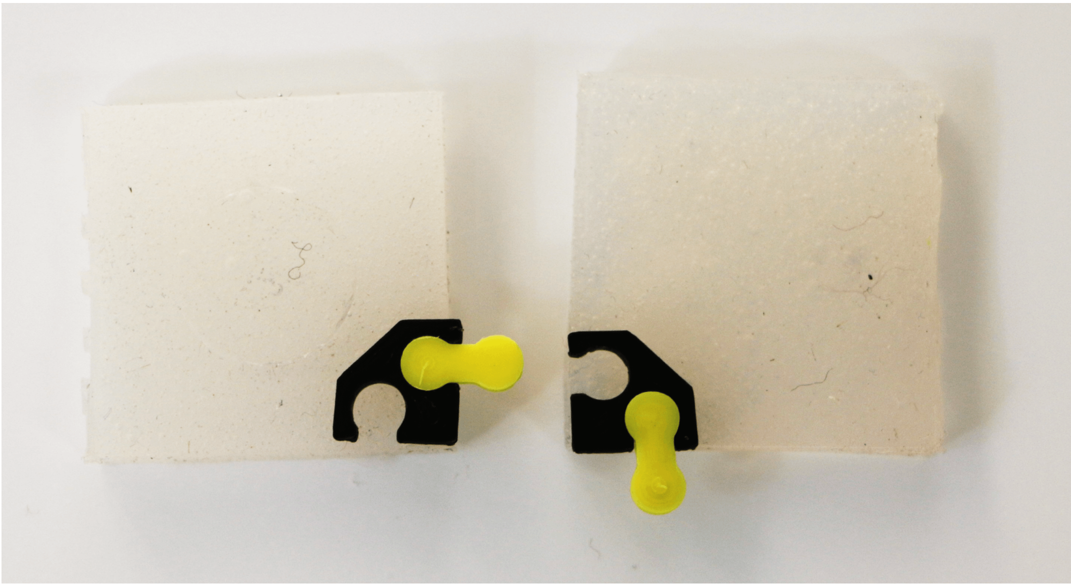


FIGURE 4.2: 3D-printed physical constraint.

**First Iteration: 3D Printed Connectors**
The first iteration of connectors make use of five pogo-pins [18] embedded in a 3D printed shape like shown in figure 4.1. This version does not snap together but provides a Poka-Yoke constraint that prevents misconnecting them. Figure 4.2 shows a 3D printed addition that holds the modules together when stretched. Future versions of the 3D printed connector could also embed magnets in order to make the connectors snap together.

However, creating tiny complex components with a 3D printer requires a high accuracy, which is difficult to achieve. Using a laser-cutter to cut connectors out of plexiglass or MDF will provide more accuracy and speed, but removes one dimension since laser-cutters can only cut vertical. This limitation, however, is no problem since the nature of the connectors allows them to be cut when they are facing up. Different thicknesses of materials determine the thickness of the connector, and additional parts can be glued or press-fitted. Also, since pogo-pins have a round head male and female connectors should have a constraint, so they align perfectly and thus preventing the sliding-off of the pogo-pins.

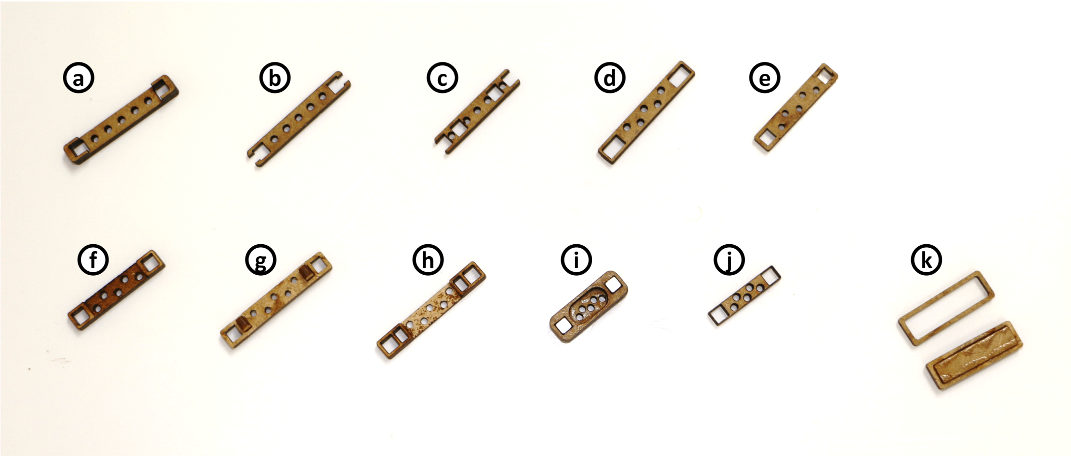



FIGURE 4.3: All iterations of connectors designs cut out of MDF. (a) straight line of pogo-pins, glued on magnet holders, (b) (c) straight line of pogo-pins, different magnet and constraint positions, (d) (e) (g) (j) trapezium shaped pogo-pin layout, different magnet and constraint positions, (f) (h) trapezium shaped pogo-pin layout, engraved central area to create a height difference, (i) trapezium shaped pogo-pin layout that also acts as constraint, (k) test with constraints.

**Second Iteration: Laser-Cut Plexiglass Connectors**

The second iteration of connectors is made out of laser-cut MDF or plexiglass and a lot of different design iterations went over them. MDF connectors are only made to test a design quickly; the final connectors are made out of plexiglass. The used pogo-pins require a material thickness of 2 mm to stick out just enough and 3 mm cubic neodymium magnets can be used to make the connectors snap together.

Figure 4.3 shows all prototyped test designs for different techniques. Figure 4.3a starts by showing a connector with the magnet holders glued on left and right. Connectors in 4.3b and 4.3c have one straight line of pogo-pins but different positions for the magnets and constraints. Figures 4.3d, 4.3e, 4.3g and 4.3j make use of a trapezium shaped pogo-pin layout to reduce the length of the connector. Magnets and constraints are placed in different positions. Connectors shown in figure 4.3f and 4.3h have a middle section that has be engraved to create a height difference. This makes that 3 mm thick connectors can be made while still providing enough space for the pogo-pin to stick out. The connector 4.3i has a trapezium pogo-pin layout that also serves as physical constraint for the pogo-pins to align properly. Figure 4.3k shows the result of a test with a full border as constraint.

Figure 4.4 shows all fabricated plexiglass connectors. Figure 4.4a shows the very first connector. Next, figure 4.4b contains a regular connector with a trapezium shaped pogo-pin layout, followed by figures 4.4c and 4.4d which have a similar connector but engraved center piece. The connector in figure 4.4e also has holes for the physical constraints. Next, figure 4.4f shows a straight-line pogo-pin layout with glued on magnet holders. The connector in figure 4.4g makes use of the magnets itself to act as physical constraints. Figures 4.4h, 4.4i and 4.4k make use as a full enclosing border as constraint. At last the connector in figure 4.4j has a center piece that acts as the constraint.

Most encountered problems with this type of connectors are a too difficult connection between male and female parts, pogo-pin misalignment in male and female connectors because of the lack of physical constraints, the size of the connectors or a too tedious

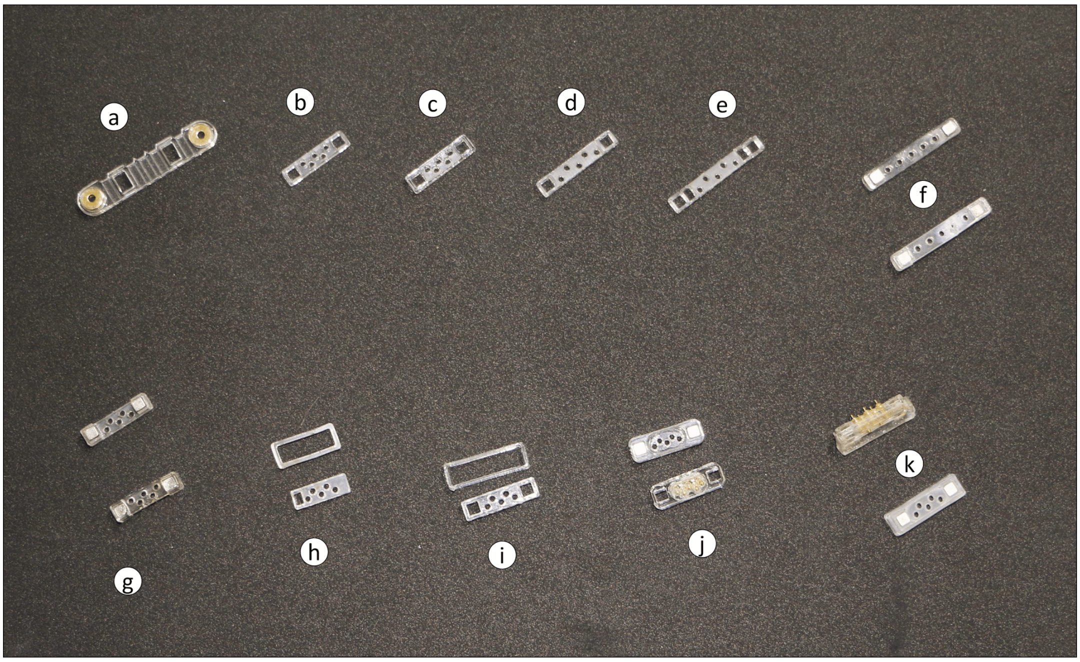



Figure 4.4: All iterations of connectors cut out of plexiglass. (a) shows the very first connector attempt, (b) regular connector with trapezium shaped pogo-pin layout, (c) (d) engraved center piece, (e) trapezium shaped pogo-pin layout with physical constraint holes, (f) straight line pogo-pin layout and glued on magnet holders, (g) trapezium shaped pogo-pin layout, glued magnet holders and magnets as physical constraint, (h) (i) (k) enclosing border as physical constraint, (j) trapezium shaped pogo-pin layout that also acts as constraint.

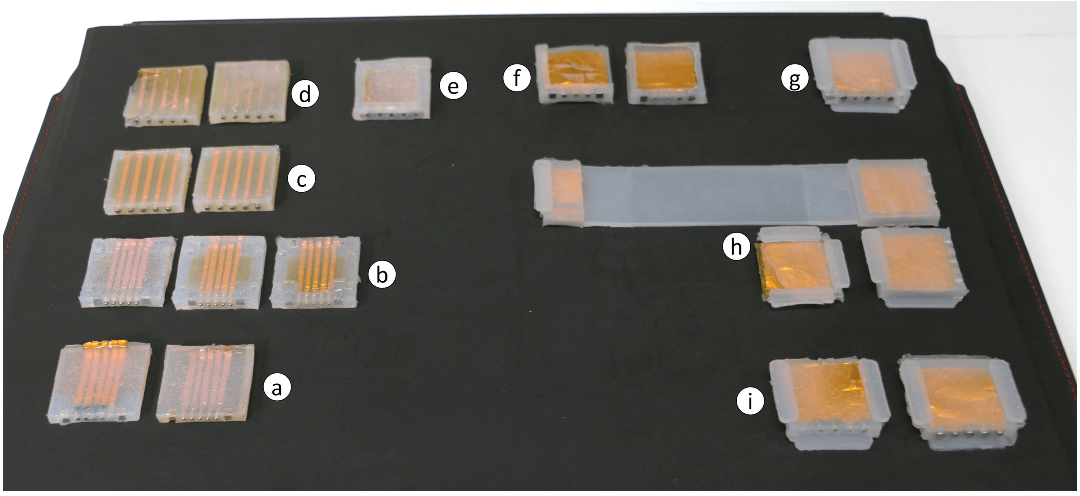



FIGURE 4.5: All silicone-only magnetic connectors with (a) the initial pogo-pin test, (b) pogo-pins with magnets fully embedded, (c) five 2 x 2 mm cylindrical magnets with silicone shore 15, (d) five 2 x 2 mm cylindrical magnets with silicone shore 5, (e) five 2 x 2 mm cylindrical magnets with a reduced silicone thickness, (f) three 2 x 2 mm cylindrical and two 3 mm cubic magnets on both sides with a reduced silicone thickness, (g) (h) pattern of 3 mm, 2 mm, 3 mm, 2 mm and 3 mm magnets with physical constraints, (i) pattern of 3 mm, 2 mm, 3 mm, 3 mm and 2 mm magnets with physical constraints.

fabrication process. Also, since the connectors are rigid, they prevent the flexibility of the modules.

### Third Iteration: Silicone Only Connectors, Pogo-Pins

The third iteration fixes the problem of the rigid connectors by a newly created design. This design uses the flexible PCB as a holder for the pogo-pins since the solder does not allow big movements. Figure 4.5 shows all created connectors. All connectors are cast in silicone as if they would be in a module. Traces of copper tape act as flexible PCB and can be measured for continuity to check if the connection between two connectors is good. The first test with this design, shown in figure 4.5a, shows that this approach as an opportunity. However, there was one big problem: since silicone does not stick to magnets, they almost immediately come loose when disconnecting two connectors. In the next iteration, the magnets were fully enclosed by the silicone, solving this problem. Figure 4.5b shows those connectors.

However, the pogo-pins shift next to each other instead of pushing each other inwards. This problem occurs because the new connectors do not contain a physical constraint that forces them only to move in one way. In combination with very tiny silicone left-overs, the pogo-pins can disconnect and thus interrupt the signal. While testing it appeared that one out of five pins (always a random one) was not connected.

### Fourth Iteration: Silicone Only Connectors, Magnets

The last iteration addresses this problem by using 2 x 2 mm cylindrical neodymium magnets as conductors. Soldering the magnets to the flexible PCB requires significant precision to maintain the spacing between magnets and respect the distance between the edge of a module and the magnets. To facilitate the soldering process, a fixture is laser-cut to precisely positioning the magnets with respect to the flexible PCB, as described in subsection 4.1.3. An additional benefit is that no additional magnets are required since the conductors already attract each other. Figure 4.5c shows the result of this technique.

While testing the created connectors, it becomes clear that the used magnets are not strong enough to hold the connectors together when they are bend. A test with the softer silicone shore 5, as shown in figure 4.5d, gives better results but still not good enough. Making the silicone shore 5 of the modules thinner, like in figure 4.5e, fixes the problem, but is because of all future electronic components not achievable.

Figure 4.5f shows the next iteration of the connectors. Replacing the two side magnets with bigger 3 mm cubic neodymium magnets creates enough attraction force to prevent the bending problem. The next problem addresses the required Poka-Yoke constraint to prevent a wrong connection of the connectors. Since silicone does not stick to plexiglass, as shown in the first connector attempt in figure 4.5g, constraints have to be made out of silicone. However, silicone constraints are stretchable, flexible, and do not slide next to other materials, which makes that they are difficult to use.

One possible attempt is to create a specific set of holes in front of the magnets that only corresponds with the correct orientation of the other connector, as shown in figure 4.5i. This technique however doesn't work because molding silicone is not accurate enough for the gaps and smaller holes are expanded by the magnets. The next attempts makes use of silicone hood on the bottom and top of the connectors. Each type of connector has the hood or on the top or on the bottom of the connector, so that opposite hoods are both on the bottom or both on the top of a module. This attempt is shown in figure 4.5h, but since the constraints are made out of silicone they just bend away and are not preventing anything. They only provide visual feedback to show that something is connected incorrectly.

The solution is to align the polarity of the magnets so that there can never be a direct short-circuit between power and ground. Following five connector points are being used: VCC (3.3V), SDA (serial data), trigger, SCL (serial clock), and GND (ground). Figure 4.6a shows the final connector design consisting of three 3 mm cubic and two 2 x 2 mm cylindrical neodymium disc magnets, with a silicone hood that should prevent connection upside down but only provides feedback. Figure 4.6b shows how the connector has a magnetic attraction to two connectors of another module and a magnetic repulsion from the two other connectors. Additionally, some of the magnetic electrodes in a connector are swapped to protect modules that are not properly aligned to snap together. All modules come with an additional hood to avoid connecting a module upside down. Even when users would enforce such inappropriate connections, the connectors are designed to prevent short circuits[1]. The spacing of the magnets and their pole layout also ensures that stretching a module cannot cause misalignment of pins.

This setup also provides enough magnetic strength to bend our soft modules up to 90 degrees without electrically disconnecting the modules (Figure 4.7a). All modules have a hardness of Shore A 5 which is one of the softest silicones. The thickness of the cable modules (3mm) allows for stretches up to 27% before the magnetic connector releases (Figure 4.7b).

### 4.1.3 Molds

Soldering magnets is a delicate process since they require significant precision and the heat of the soldering iron can change or remove the polarity of magnets. However, by

[1] This feature however is not yet implemented in the current design because of a mistake.

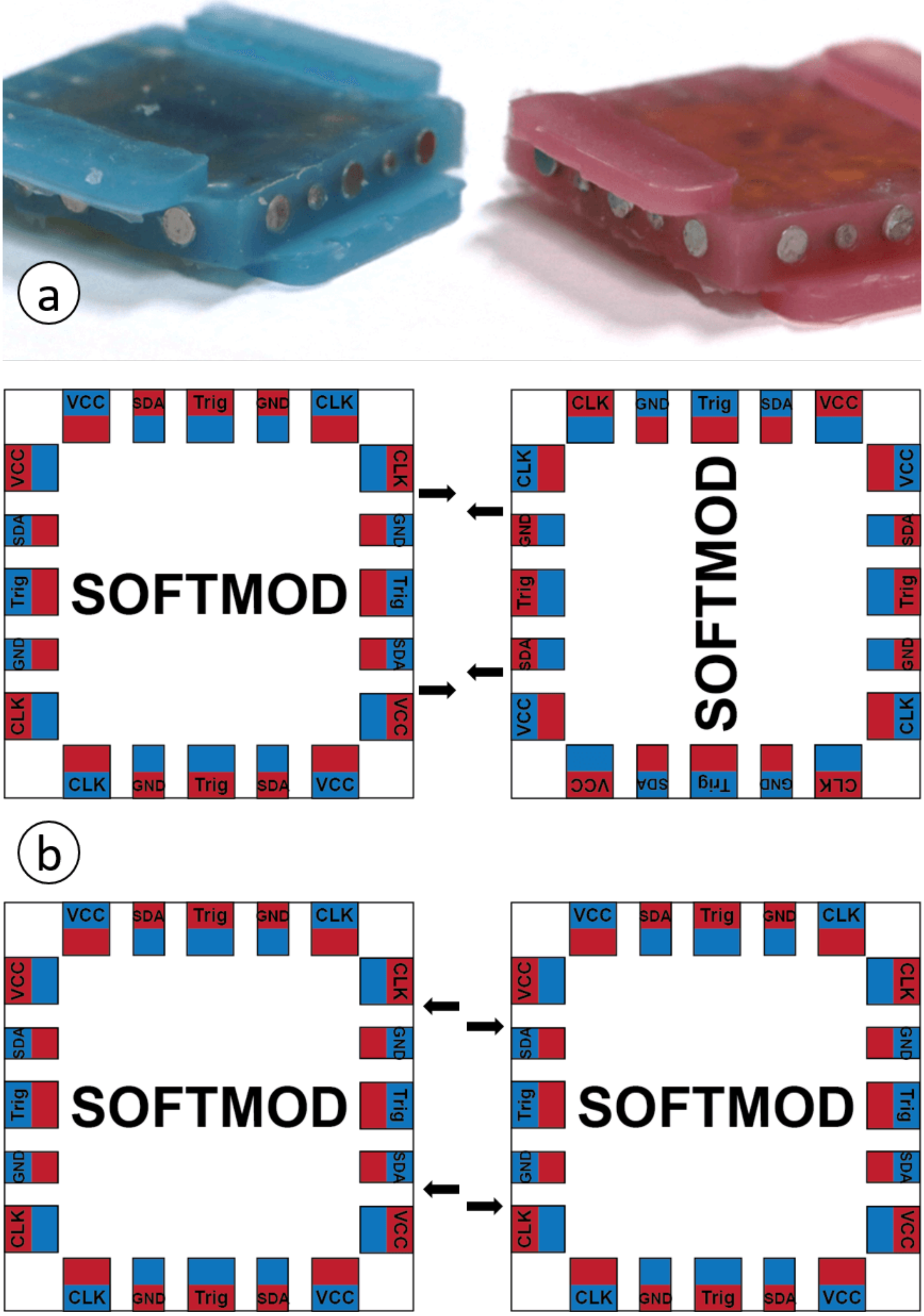



FIGURE 4.6: (a) SoftMod's magnetic connectors embed magnets that (b) attract two connectors of another module and repel from two other connectors of that module.

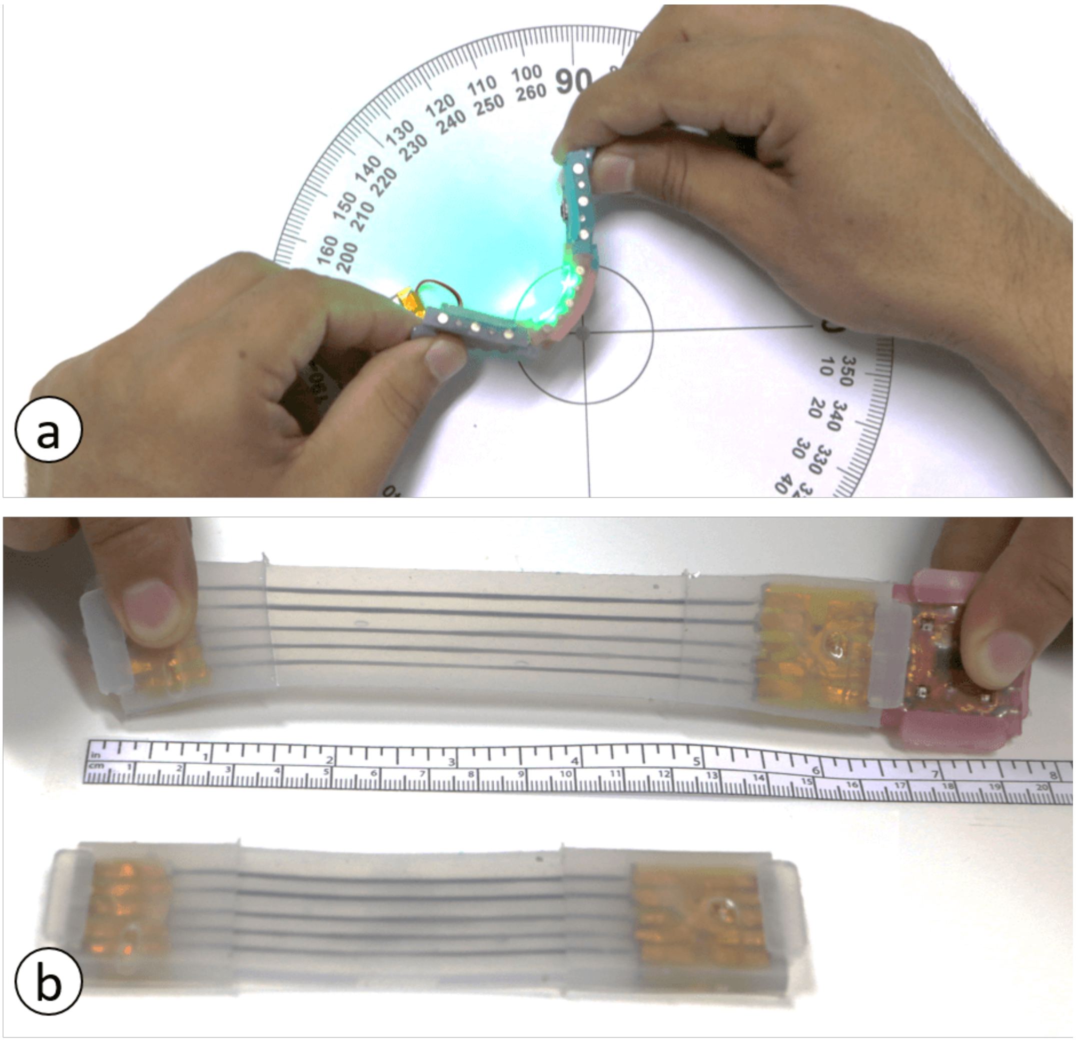



Figure 4.7: The magnetic connectors are strong to ensure electrical conductivity while (a) bending and (b) stretching modules.

placing a big magnet underneath the to-be-soldered magnet, the change in polarity will not occur. An MDF mold simplifies the soldering process by holding the magnets in place in front of the big magnet. The mold works by first loading a small magnet into the holder, placing the flexible PCB mounted on a piece of plexiglass in front of it and finally soldering the magnet to the PCB. For the next magnet, the PCB has to shift one hole to left or right, so the to-be-soldered magnet is always in the center hole. Several versions of the mold are used for different magnet polarities and sizes. In total, four molds are constructed: one positive and negative for 3 mm magnets, and one positive and negative for 2 mm magnets.

Soldering magnets on one side of the PCB is easy with the mold, but not when there are already magnets on the other. This difficulty is because magnets always stick to everything else and once soldered they release fast because of the magnetic soldering iron. A new mold is created to improve the soldering process, where all magnets are loaded and soldered at once. Figure 4.8 shows this mold. The to-be-soldered magnets are held in place by an MDF piece and magnets sticking in the borders. The downside of this mold is that the magnets in the wall are a little bit too small to prevent the depolarization, so fast soldering is required.

## 4.2 Fabrication of Soft Modules

Initially, the goal was to make all modules stretchable using stretchable soft silicone and liquid conductors (i.e., Galinstan), according to the procedure of Nagels et al. [51]. However, as this is mostly a manual process, and most electronic components (like the connection pads of the ATtiny1614) were too tiny, it was too tedious and error-prone for making a set of modules using only the described procedure. The first subsection 4.2.1 describes the design iterations that went over the creation of the soft modules. Next, subsection 4.2.2 shows the development of the stretchable wires based on the technique developed for the soft modules. Finally subsection 4.2.3 goes deeper in on the details of the created molds.

### 4.2.1 Soft Modules

Following paragraphs describes the design iterations of the soft modules. As mentioned in the introduction of this chapter, all PCB related information is described in chapter 5.

**First Iteration: Copper Extension Pad and Galinstan**
One possible solution is to extend the pads of components by creating copper tape extensions. Making the extensions can be done by, for example, a fiber laser-cutter or vinyl-cutter. When using Galinstan as conductor for all traces modules are still stretchable, except for the parts where the extension pads are. To be able to create the full circuit, there are two layers of Galinstan traces required with multiple VIAs to connect them. The enclosure is made out of silicone 15 and has dimensions 60 by 60 by 5 mm. The first layer of the enclosure is a silicon layer of 1.5 mm thick. Next, the extension pads are placed, and another 1 mm silicone layer is added. The third layer contains the Galinstan traces of the bottom half of the circuit, followed by another layer of 1 mm silicone. Finally, the top half of the circuit is created in Galinstan together with all VIAs and filled with another 1.5 mm silicone layer. Figure 4.9 shows a module created by this technique.

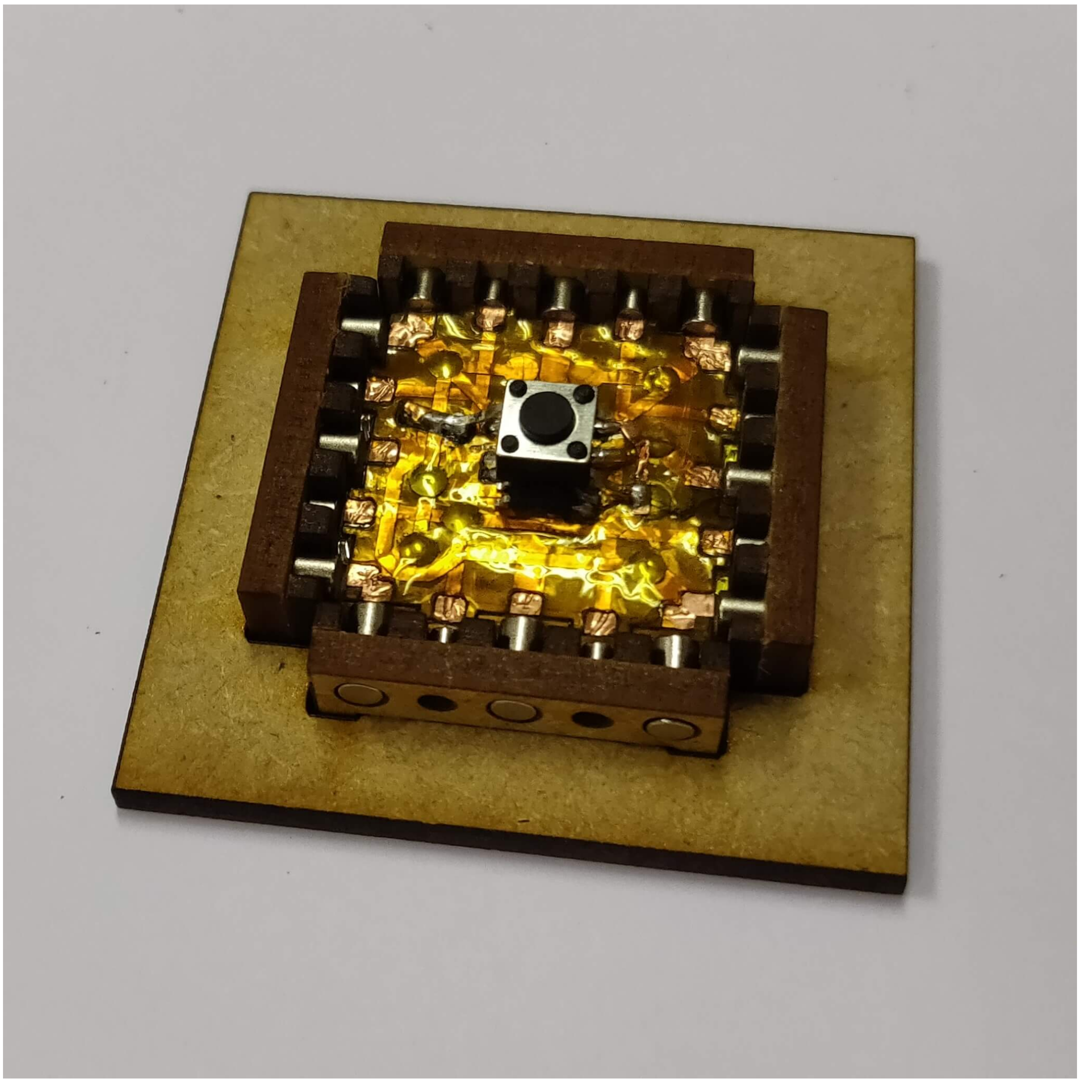
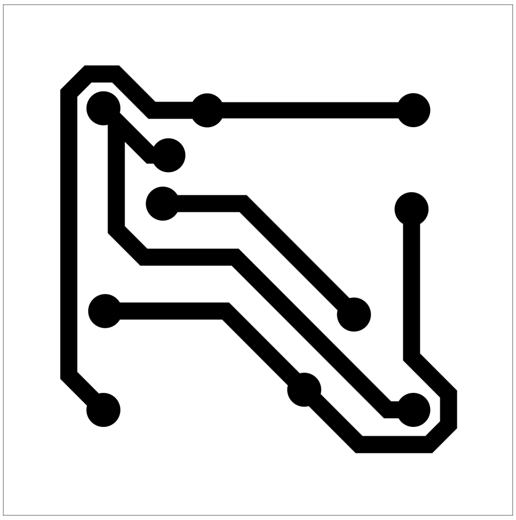
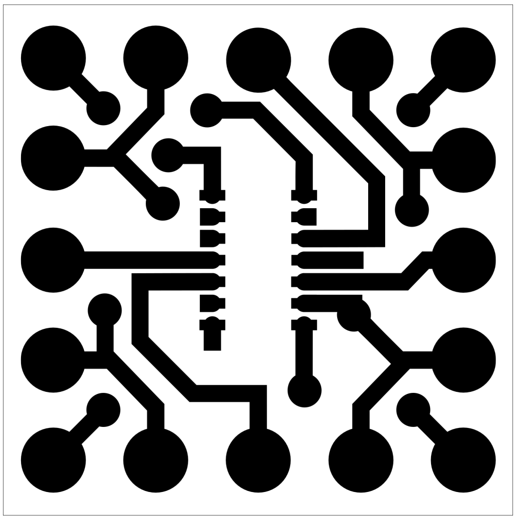


FIGURE 4.8: Second version of the mold to solder magnets, mostly used when soldering magnets on all four sided of a PCB.

FIGURE 4.9: First iteration of the fabrication of modules using a copper extension pad and Galinstan.



(A) Bottom layer circuit.



(B) Top layer circuit.

FIGURE 4.10: Design of the first iteration of the flexible PCB.

However, creating modules with this technique was still too tedious since there are too many traces and VIAs. Also, it is not easy to pore silicone in very thin layers since components that are sticking out cause unevenness. Another flaw in the design becomes apparent when stretching the modules. Since the copper tape is not strong enough, it immediately snaps when it gets pulled. Besides using a harder silicone multiple layers of copper tape or some other kind of reinforcement can fix this problem.

**Second Iteration: Flexible PCBs and Galinstan**

Addressing the problems of the first iteration, a new technique to create flexible PCBs was designed, consisting of cutting copper tape and Kapton with a fiber and CO2 laser, respectively. Chapter 5 goes deeper in on the details of this technique. Figure 4.10 shows the top and bottom layer of the created circuit, figure 4.11 displays a finished circuit. The flexible PCB contains all bridges and VIAs needed to interconnect all components, causing everything can be soldered easily. Connectors are then connected to pads on the PCB by traces of Galinstan, causing the module to be still a little bit stretchable. This technique makes use of three silicone shore 15 layers, with the PCB between layers one and two and the Galinstan between layers two and three. The first layer of silicone is 2 mm thick, the next one 1 mm and finally the last one 2 mm, resulting in a total dimension of (60 by 60 by 5 mm). Figure 4.12 shows the result of the second iteration of the modules.

Although this method looks more promising, there are still some issues to fix. First of all, the module is still quite hard to fabricate since using Galinstan requires a delicate manufacturing process. Thick components cause bumps and inequalities in the silicone and connectors create gaps, as seen in figure 4.13. Because of the required multiple layers of silicone, this process is still really error-prone. Also, since most elements are already non-stretchable, the module can only be stretched in very few areas. Due to the size of a module, some smaller structures like a bracelet are less useful since those assemblies won't contain enough modules and thus functionality for the available space.

**Third Iteration: Flexible PCBs and small modules**

In the third iteration, the size of the modules is reduced to 30 by 30 by 5mm. This

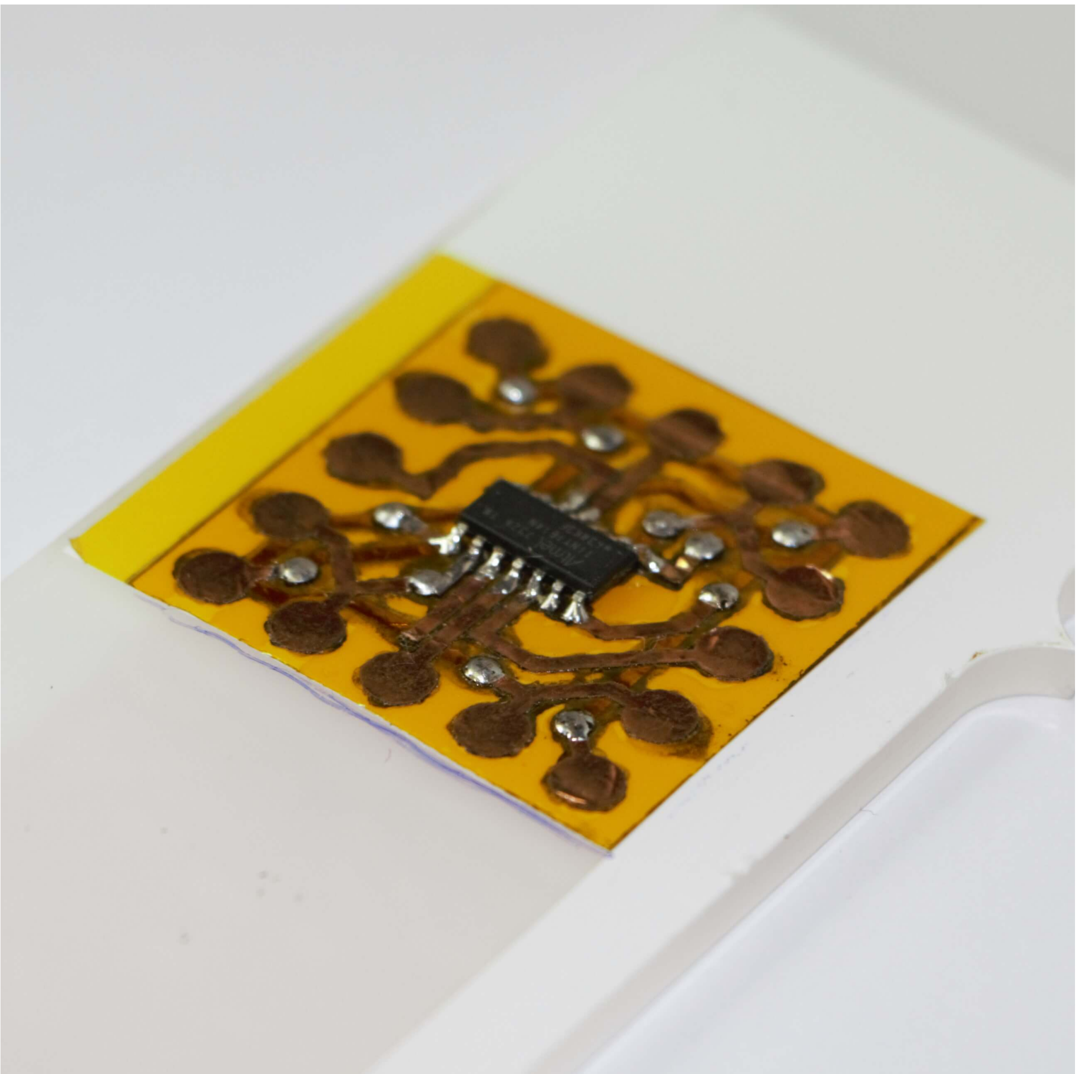


Figure 4.11: Result of the first iteration of the flexible PCB.

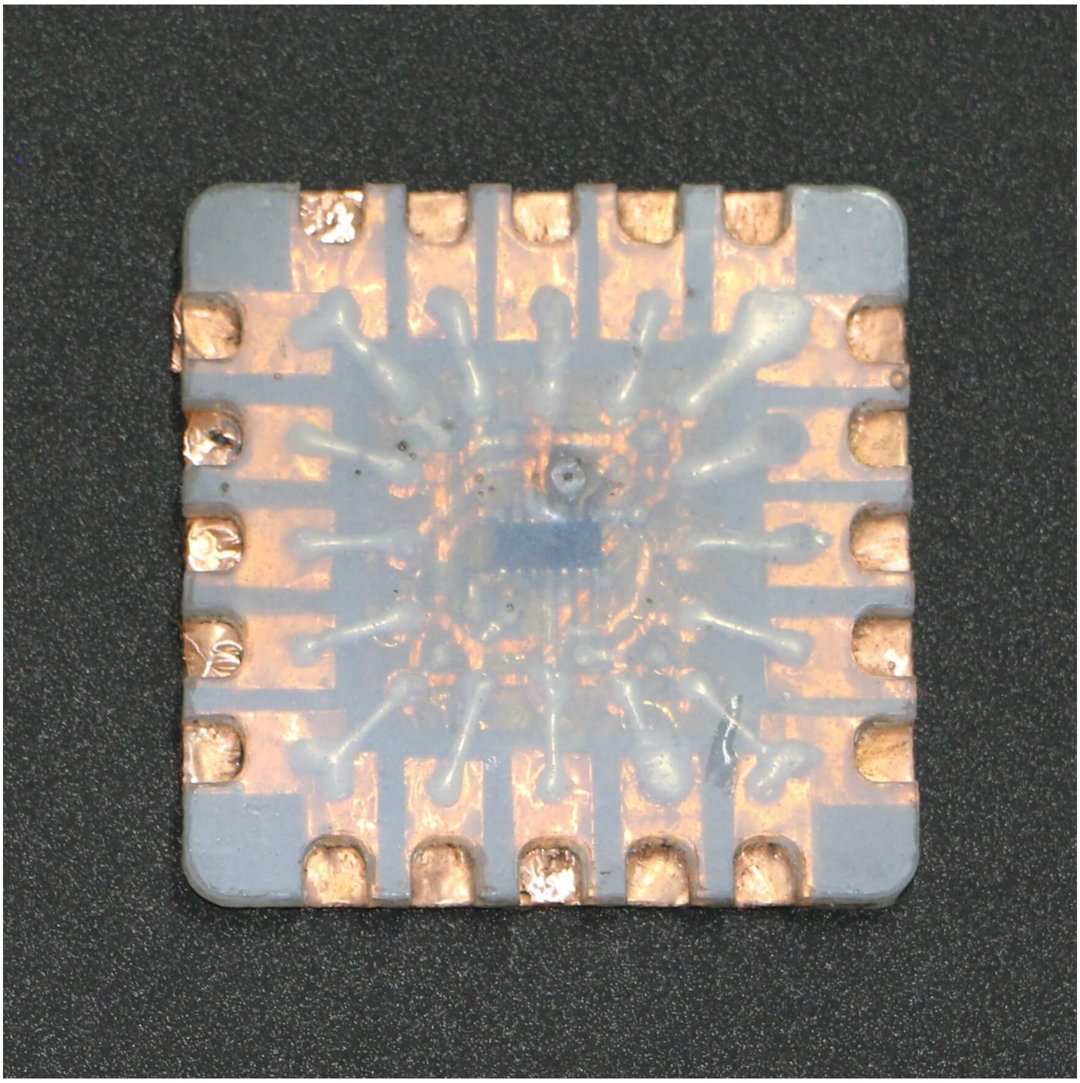


Figure 4.12: Second iteration of the fabrication of modules using a flexible PCB and Galinstan.

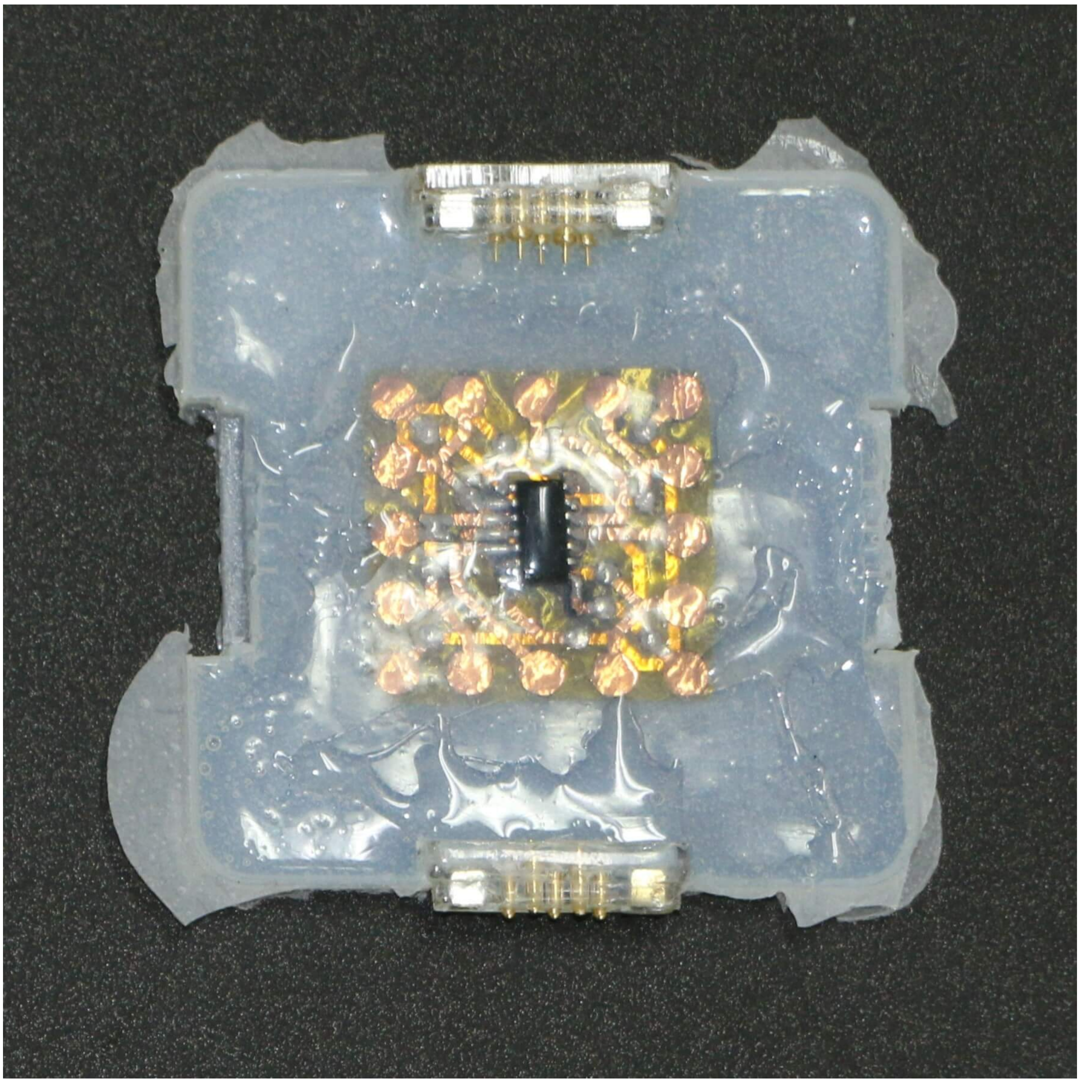


FIGURE 4.13: Second iteration of the fabrication of modules. Thick components like the ATtiny1614 cause bumps in the silicone, and connectors are causing gaps and thus are not usable with Galinstan.

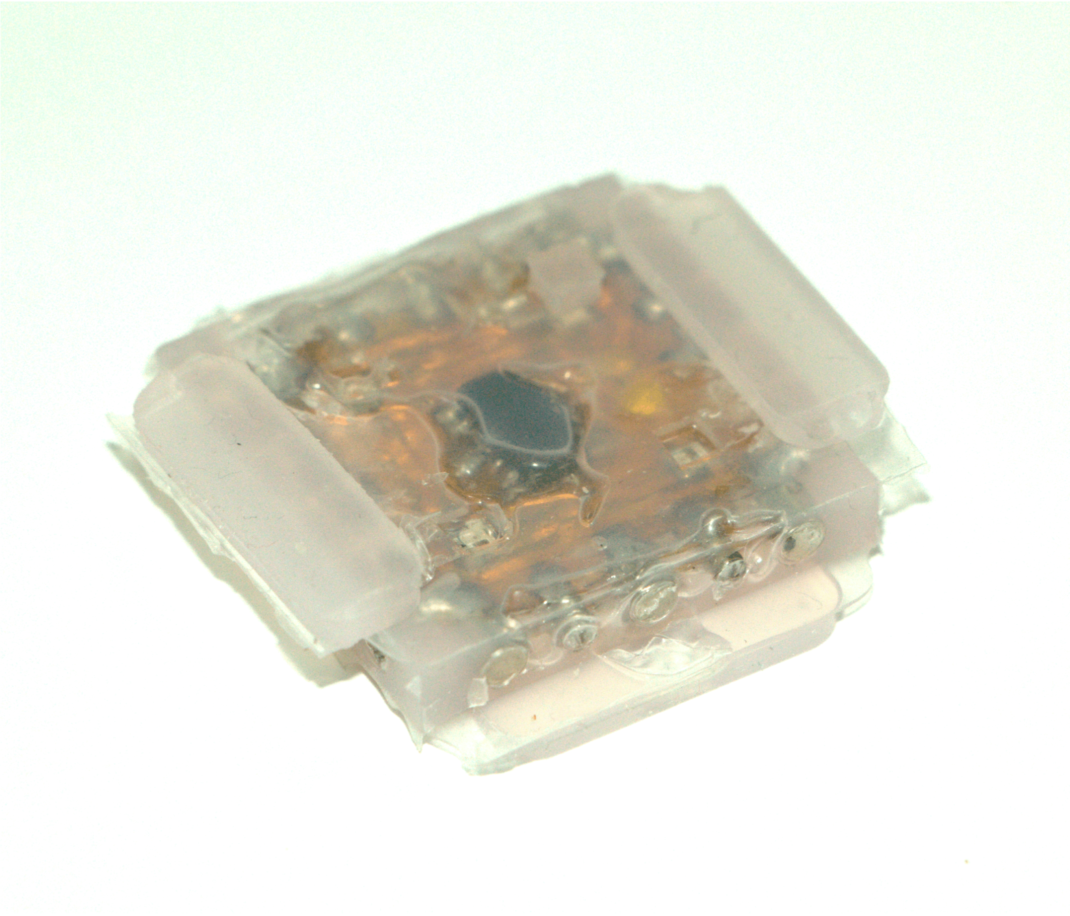
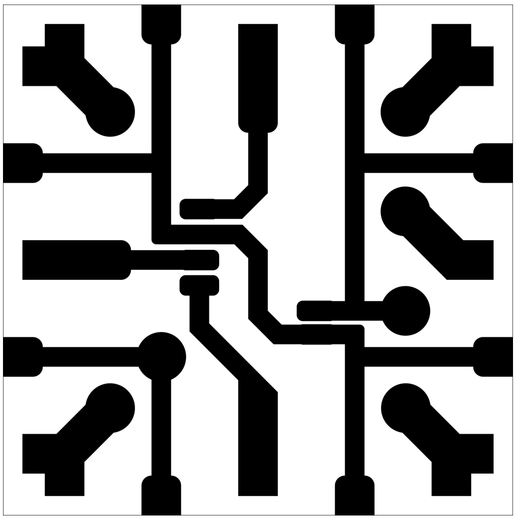


FIGURE 4.14: Third iteration of the fabrication of modules using a flexible PCB and reducing the size of the modules.

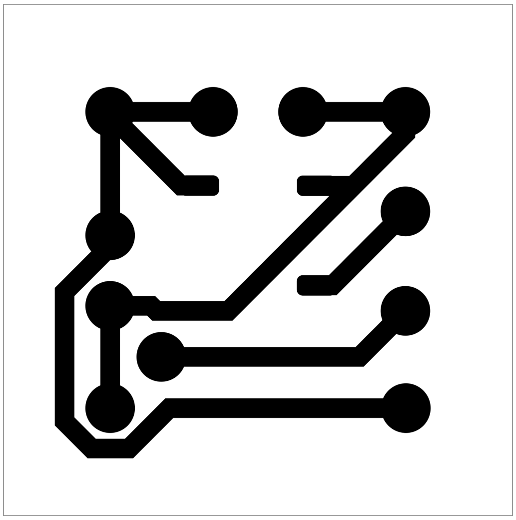
reduction in size means that connectors are soldered directly to the flexible PCB without any Galinstan traces. However, this means the flexible PCB has to change to fit the connectors. Also, the process of creating silicone modules simplifies because now it only takes one layer of silicone shore 15, which allows the use of molds for pouring silicone. This molds, described in subsection 4.2.3, holds the flexible PCB by its magnets in the air what makes that silicone can reach underneath the PCB. Figure 4.14 shows a finished module.

Figure 4.15 shows the designs of the new circuit for the RGB LED module. The first two circuit layers are always the same for all modules, the third layer depends on the functionality of the module. Another change is that the pads for the ATtiny function as vias, which reduces the complexity of the circuit and allows access to the microcontroller on all layers. Figure 4.16 shows the finished circuit for the RGB LED module.

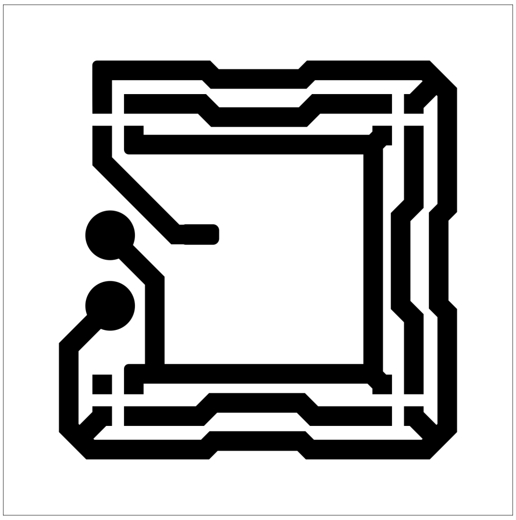
This technique solves the problem of the too big modules but also introduces new issues. Because of the use of magnets as conductors, the traces of the flexible PCB are not strong enough to hold the force of the attractions, resulting in snapping copper traces. Applying a silicone glue before casting the PCB into silicone fixes this problem, but also creates them. First of all, the interconnectivity of the magnetic conductors is not good because the magnets require or a very high precision alignment or a little bit moving space. Since this approach is entirely DIY that high precision can not be achieved, and because of the glue, the magnets can not move anymore. Also, the



(A) Bottom layer circuit.

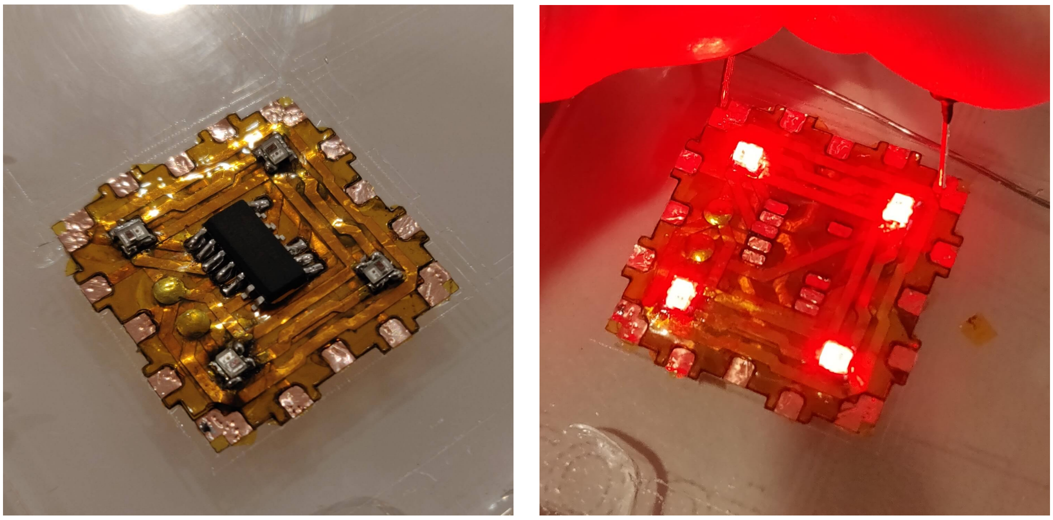


(B) Top layer circuit.



(C) Dynamic third layer depending on the functionality of the module.

FIGURE 4.15: Design of the second iteration of the flexible PCB.



(A) All components soldered. (B) RGB led test.

FIGURE 4.16: Result of the second iteration of the flexible PCB.

adhesive prevents the silicone shore 15 to reach everywhere, causing big air bubbles as noticed in figure 4.14. This problem, however, can be fixed easily by using a silicone shore 5, which has a lower viscosity, is more liquid and also more flexible after curing. By filling the entire mold with silicone and then placing the flexible PCB on the magnets in the frame, air bubbles are avoided. Figure 4.17 shows the finished result of the modules.

### 4.2.2 Stretchable Wires

Since wires are relatively simple in comparison to other modules, they are made entirely stretchable. Using Galinstan as a liquid conductor does not give the problems described in subsection 4.2, since wires only contain connectors on two sides, instead of four. This simplification allows soldering both connectors on to two opposite flexible PCBs that are interconnected using five traces of Galinstan. By using a similar mold to that one of the modules and two layers of 1.5 mm silicone shore 5, stretchable wires can be created easily. First, the mold has to be filled in half with silicone. Next, the flexible PCB, connectors and Galinstan traces are placed, followed by a final layer of silicone to encapsulate all components.

### 4.2.3 Molds

Using molds with silicone improves the process of creating soft modules. Molds are made out of stacked layers of laser-cut plexiglass, inspired by StackMold [69]. Specific heights in the molds are achieved by using different layer thicknesses, or by custom-created scraping tools. Small MDF joints hold the layers together by press-fitting them into the made holes. An issue with this type of molds is that silicone shore 5 leaks between the layers of plexiglass causing silicone flaps hanging on the side of the modules and thus requiring an extra cleaning step. Figure 4.18 shows the test results of all created molds. The two most important iterations are described by the following paragraphs.

**Molds for Simple Square Modules**

The first iteration of molds produces simple square-shaped modules with dimensions

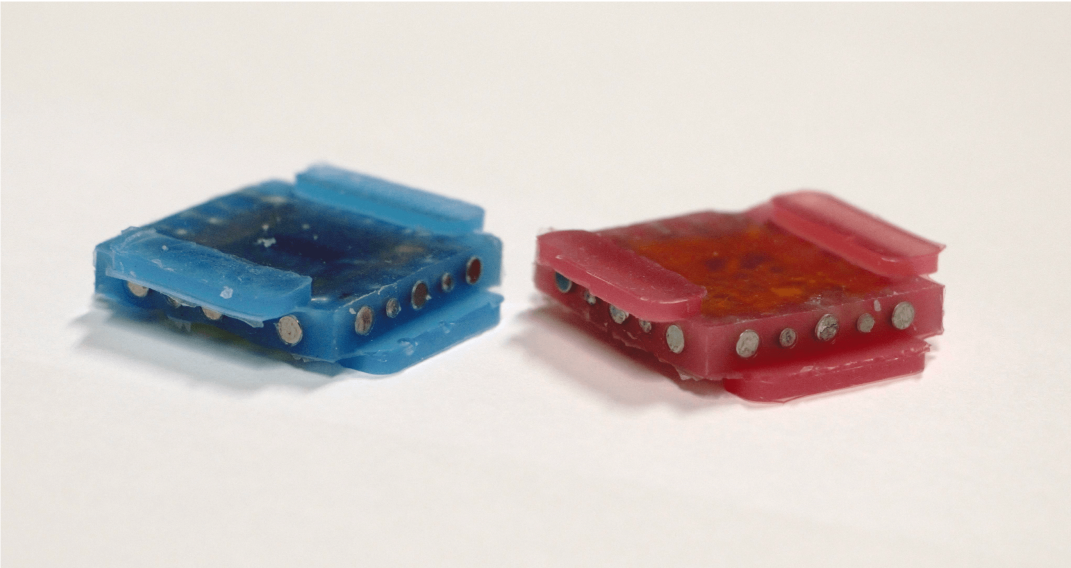


FIGURE 4.17: Two finished modules created with a flexible PCB and mold for the silicone.

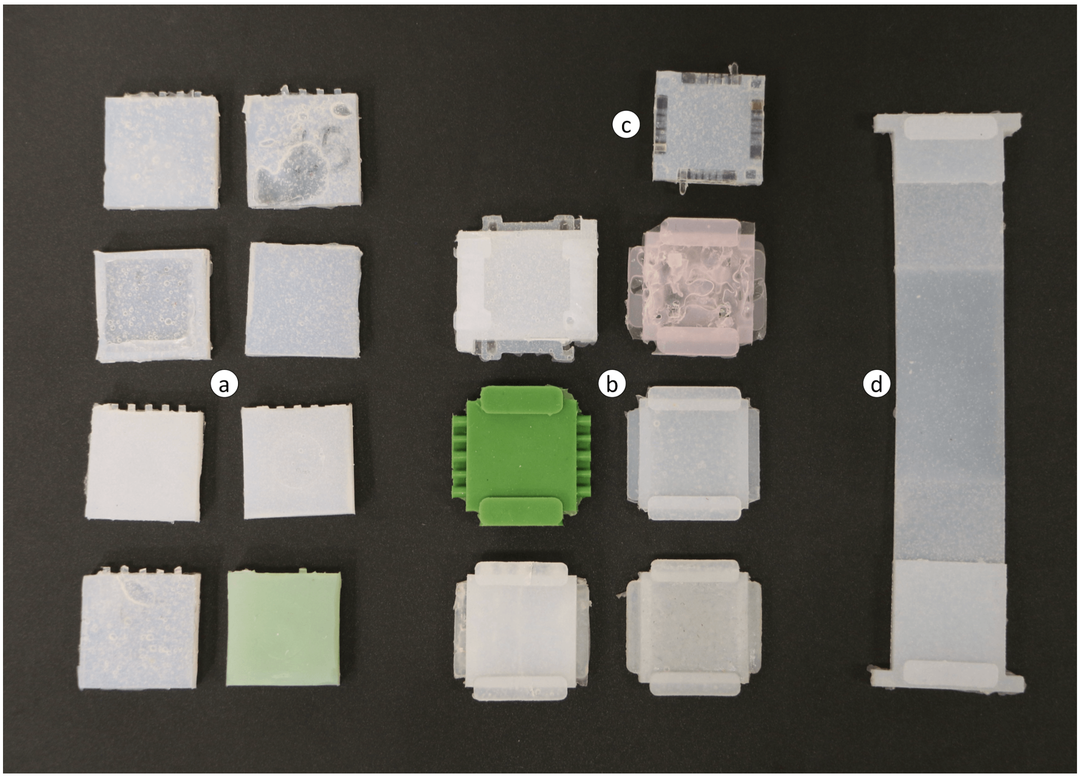



FIGURE 4.18: All iterations of the created molds with (a) showing the results of the first iteration, (b) the results of the second iteration, (c) an experimental design and (d) the first test of the wire mold.

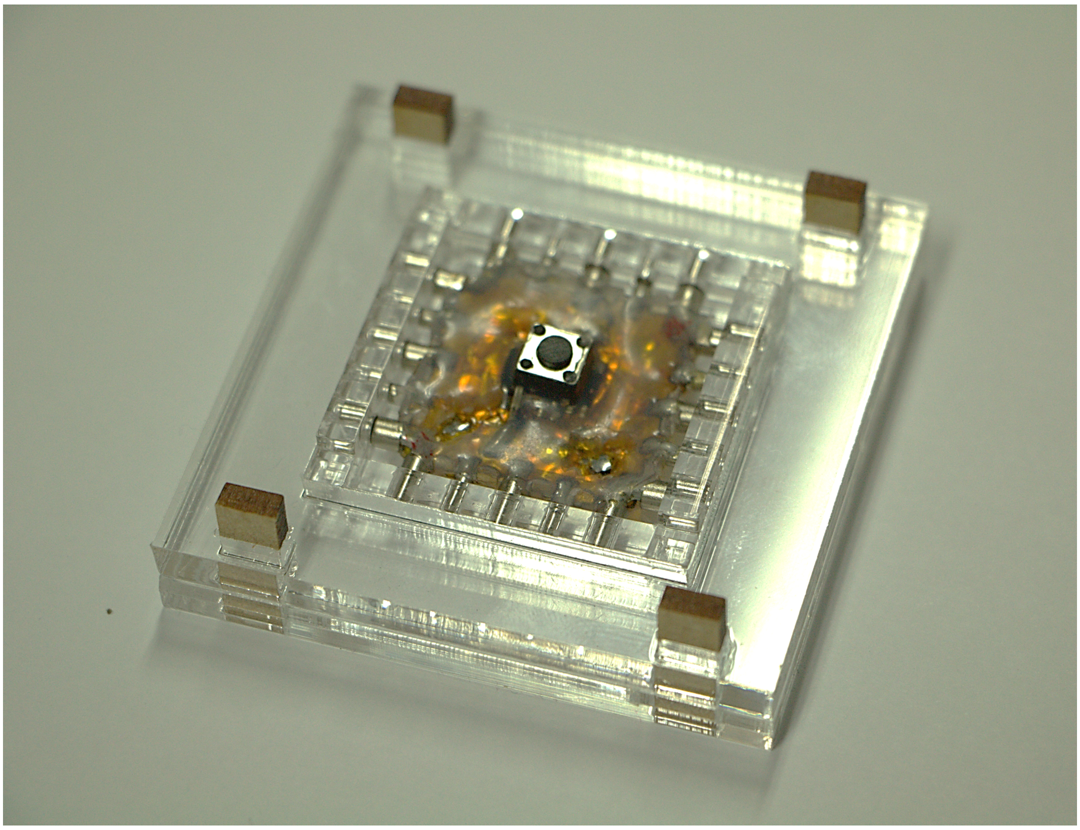


FIGURE 4.19: First iteration of the mold used to create silicone modules.

30 by 30 by 5 mm. On the bottom of the mold, there is one full layer of 3 mm plexiglass. Next, the mold contains a layer of 5 mm with a 36 by 36 mm rectangle cut out. Four borders with magnets, created out of 3 mm plexiglass and placed on their side, keep the flexible PCB in place and prevents the silicone from flowing over the end of the connectors. Figure 4.18a shows the first eight test samples created with this mold, with varying silicone hardness, color, and thickness. Figure 4.18c is an experimental design with a plexiglass constraint. Figure 4.19 shows this mold with a flexible PCB placed inside.

**Molds for Modules with Physical Constraints**

The second iteration of molds also includes the space for the physical constraints described by section 4.1. The mold starts with a full layer of 3 mm plexiglass. Next, a 2 mm layer is added with cut-out holes which create silicone constraints on the bottom of the module. The next layer of the mold is a 5 mm thick plexiglass part with a 38 by 38 mm rectangle cut out in the center. This layer contains the four borders with magnets created out of 4 mm thick plexiglass. These holders need to be at least 4 mm thick to cover the holes created for the constraints, so they remove holes the silicone can flow into. A 2 mm layer with the holes for the constraints for the top of the module finishes the mold. Silicone can then be poured into the holes of the top layer. The magnet borders are held in place by holes cut in the 2 mm top- and bottom layers, on the opposite side of the holes for the constraints. Figure 4.18b shows the results of six samples created with this mold. These samples all have different silicone hardness and colors. Because silicone shore 5 is the most liquid it is recommended to

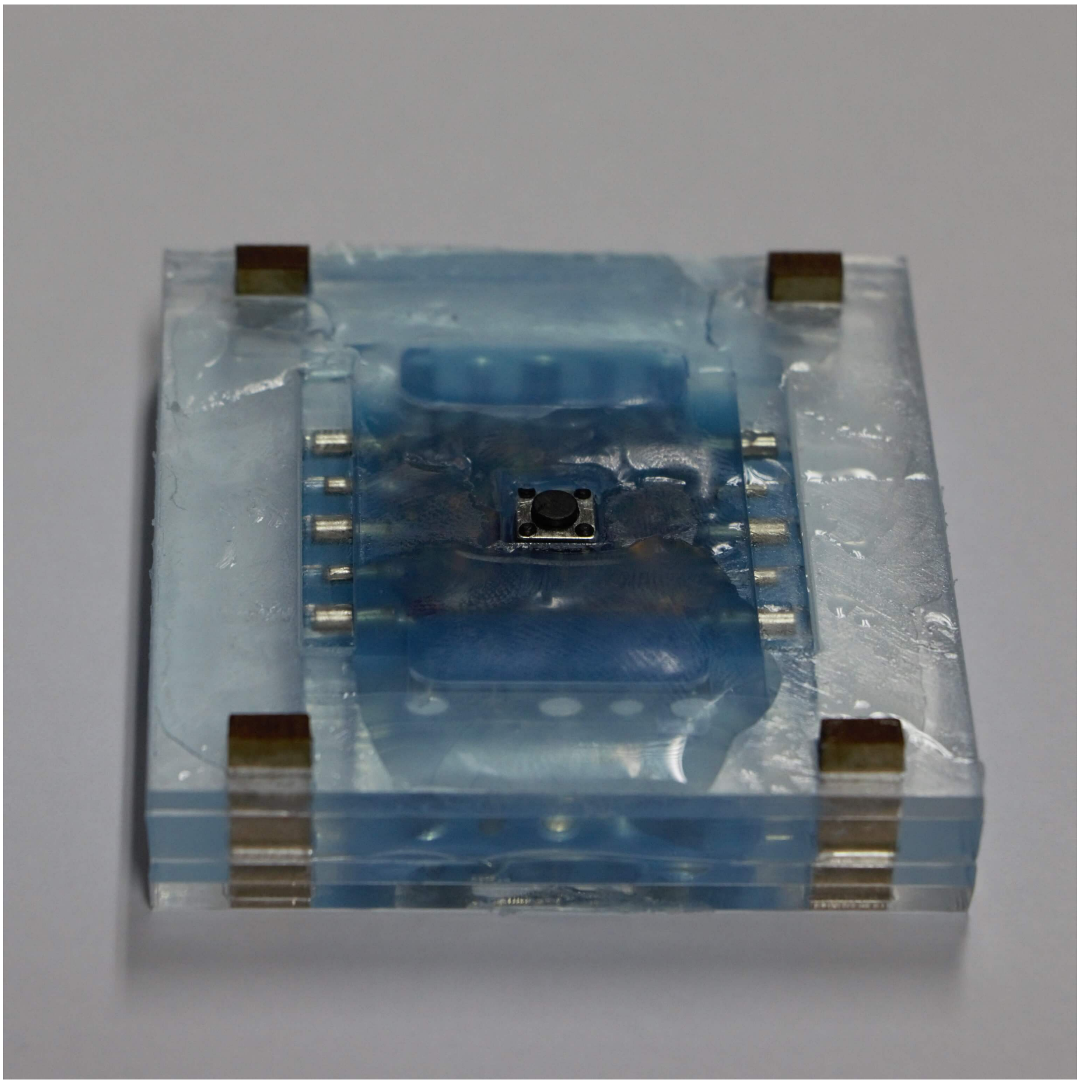


FIGURE 4.20: Second iteration of the mold used to create silicone modules with physical constraints.

use this one. Figure 4.20 shows the mold with a flexible PCB placed inside, cast in blue silicone.

**Molds for Wires**

Since all modules except one are the same, only two types of molds are required. This paragraph describes the mold for the wires, which builds further on the mold for the module with physical constraints. The main difference is the number of connectors and the length of the mold. This mold consists of three parts: the main part with connector and flexible PCB, the stretchable center part that contains the Galinstan and the last part that only contains a connector. The first and third parts are 5 mm thick, the center part only 3 mm. This mold is also created by stacking several layers of laser-cut plexiglass on top of each other, hold together by press-fit MDF joints. The first layer is a 3 mm layer that serves as a base plate. Next, two pieces of 2 mm plexiglass with the holes for the constraints cut out are placed on the first and third parts. A 3 mm piece is placed in the middle section, causing the required height difference of 1 mm. Next, the first and third part contains a 5 mm piece with the PCB and connectors cut out, together with two magnet holders for the connectors.

Finally, a full 1 mm piece of silicone is placed on top of the middle section creating another height difference of 1 mm in the top layer of silicone. Figure 4.18d shows the result of the first test.

# Chapter 5

# Fabrication: Flexible PCB's

The second part of the fabrication process is described in this chapter, which explains the full process of a novel DIY technique created to be able to fabricate flexible PCB's. This technique makes use of the combination of a CO2 and fiber laser to cut Kapton and copper, respectively. A poster about this technique has been submitted to UIST 2019 ([39]).

This chapter describes the created technique independently from SoftMod since it can be used universally. However, since this technique is created for SoftMod all images will come from there. First, section 5.1 gives starts with an introduction about the context of the new method. Next, section 5.2 shows the steps required to create a multi-layered flexible PCB. Section 5.3 explains the process of the calibration of the laser followed section 5.4 that conducts a functional analysis of the created PCB. Finally, section 5.5 concludes this chapter.

## 5.1 Introduction

Recently, there is an increasing interest in making flexible circuit boards using DIY equipment in Makerspaces and Fablabs [51, 72]. These techniques empower interaction designers to make functional prototypes, allow for fast design iterations, and encourage experimentation with new interface ideas, such as interactive textiles and skin [28].

Similar to popular prototyping procedures for rigid PCBs, makers experimented with DIY chemical processes, such as etching copper on flexible materials[1] and selective evaporation of copper on flexible materials using stencils[2]. Although these approaches result in highly conductive copper traces, they require advanced knowledge and careful treatment of chemicals. The approach presented by Perumal et al. [16] makes prototyping flexible copper circuits more convenient but requires a special layered material. Alternatively, vinyl cutters are used to cut circuits directly from copper sheets [27, 61]. These techniques are however limited to single-layered designs and restricted to traces of at least a few millimeters wide [61].

Over the past few years, a significant portion of research in personal fabrication focused on accessible techniques for depositing conductors on flexible substrates, such as silver nanoparticle ink [35] or PEDOT:PSS [73] conductive inks. Alternatively, CO2 laser cutters can cut circuit traces from carbon-filled silicone [72] or plastics coated with conductive ink [27]. Lin et al. [76] demonstrated laser patterning Kapton to introduce graphene with electrical conductivity. These research efforts, however, use carbon or ink-based conductors with a high resistance as highly conductive metals, such as copper, can not be processed with a CO2 laser cutter.

[1] https://www.kobakant.at/DIY/?p=240
[2] https://www.youtube.com/watch?v=FYgIuc-VqHE

## 5.2 Laser Cutting Fully Functional Circuits

The new approach, shown in Figure 5.1, builds up a stack of interleaved layers of thin-film Kapton and copper tape. The first layer consists of a layer of copper tape. Next, circuit traces are laser cut with the fiber laser (figure 5.1a) and the remaining copper is removed using tweezers (figure 5.1b). Next, a layer of Kapton tape covers the traces, and the holes of the (buried) VIAs are cut using the CO2 laser. The cut-out centers then are removed, and the holes are cleaned (figure 5.1c and figure 5.2). Afterward, a new layer of copper tape is placed on top of the circuit, and the second layer of the circuit is cut out using the fiber laser again (figure 5.1d). The remaining copper and the center of the VIAs holes are then removed using tweezers (figure 5.3). The resulting holes are filled with solder paste and cured by engraving the region with the CO2 laser (figure 5.1e and figure 5.4). A similar technique can be used for soldering components to the top circuit layer (figure 5.5. This stacking process of copper and Kapton can be iterated multiple times to create multi-layers flexible PCBs. The finished PCB looks like figure 5.6.

## 5.3 Laser Calibration

To precisely cut copper without interfering the Kapton below, the power of the fiber laser needs to be calibrated. Although a fiber laser can not cut Kapton, it is recommended to do the calibration step because if the Kapton gets too hot it burns away. The calibration process starts with laser cutting a circle inside a slightly larger square. The power is increased until the square can be peeled off while the inner circle remains attached. Figure 5.3 shows this calibration process. The CO2 laser does not require calibration as this laser does not affect the copper below the Kapton. However, it is still recommended because if the CO2 laser burns the Kapton too much, a black residue is left. This residue makes the soldering process more difficult since it prevents solder paste from flowing in the VIAs.

## 5.4 Functional Analysis

The first test is to examine the durability of the new technique. A robot arm repeatedly flexed 60 circuit traces (20 traces of 0.5mm, 1mm, and 2mm), as shown in figure 5.9. Three 0.5mm traces broke after respectively 2000, 2500, and 3000 cycles. All other traces still worked after 5000 cycles.

The next test investigates if it is possible to create a fully functional PCBs with this technique. A flexible PCB with a resistor, red LED and ATtiny1614 programmed with a basic "blink" code is designed in order to execute this test. Figure 5.9 shows the working PCB. This test concludes the functionality of the DIY created PCB.

## 5.5 Conclusion

This work contributes to a steady stream of research on using laser cutters as all-around fabrication machines [27, 50, 51, 68, 76]. While continuing the research efforts in this direction, this work can reveal a new potential of laser machines. This could trigger more research and justify acquiring fiber laser cutters in maker labs.

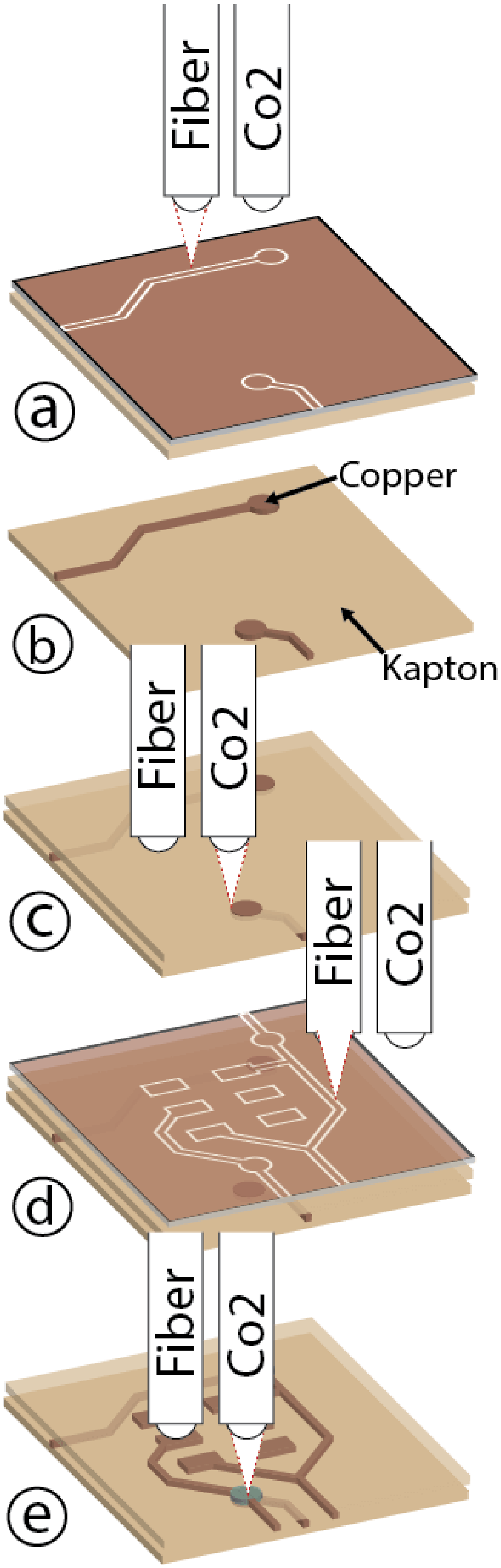



FIGURE 5.1: (a) Cutting copper traces first circuit layer, (b) removing excess copper (c) cutting VIAs from second layer of Kapton, (d) cutting circuit traces second circuit layer, (e) curing VIAs.

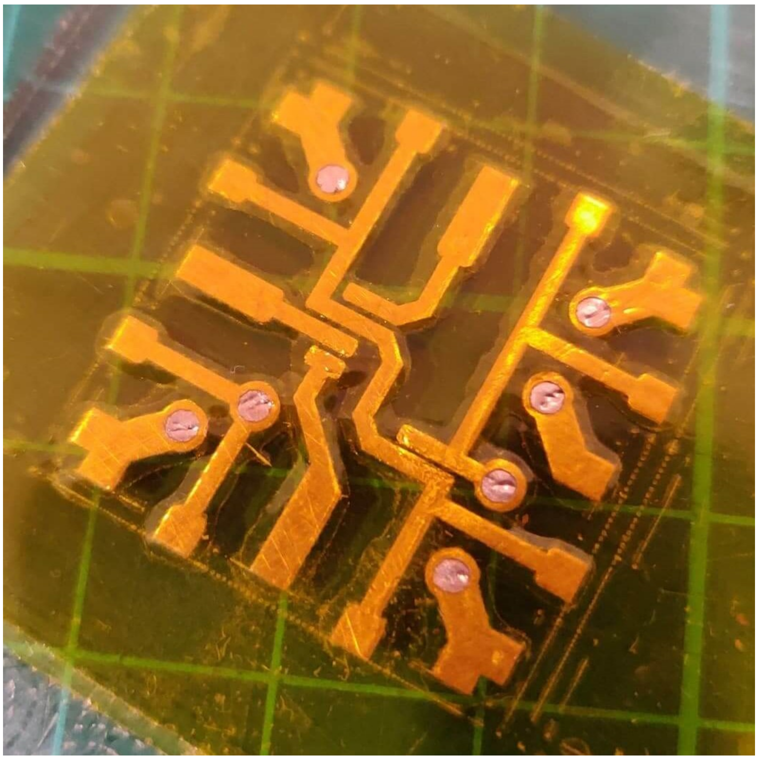


FIGURE 5.2: First layer of Kapton tape placed over the first circuit layer. The holes for the VIAs are cut out in the Kapton.

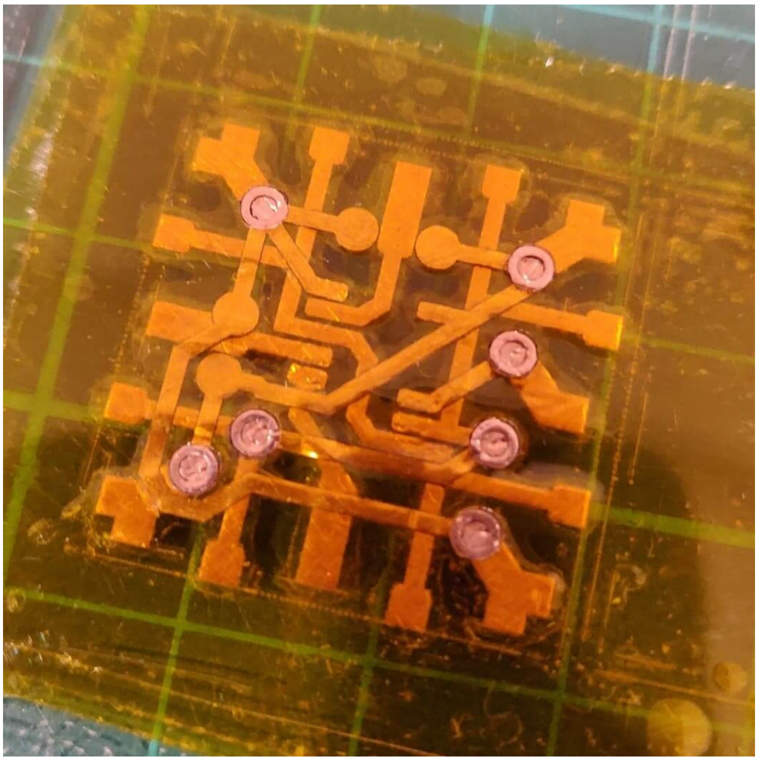


FIGURE 5.3: Second layer of Kapton tape placed over the second circuit layer. The holes for the VIAs are cut out in the Kapton and copper.

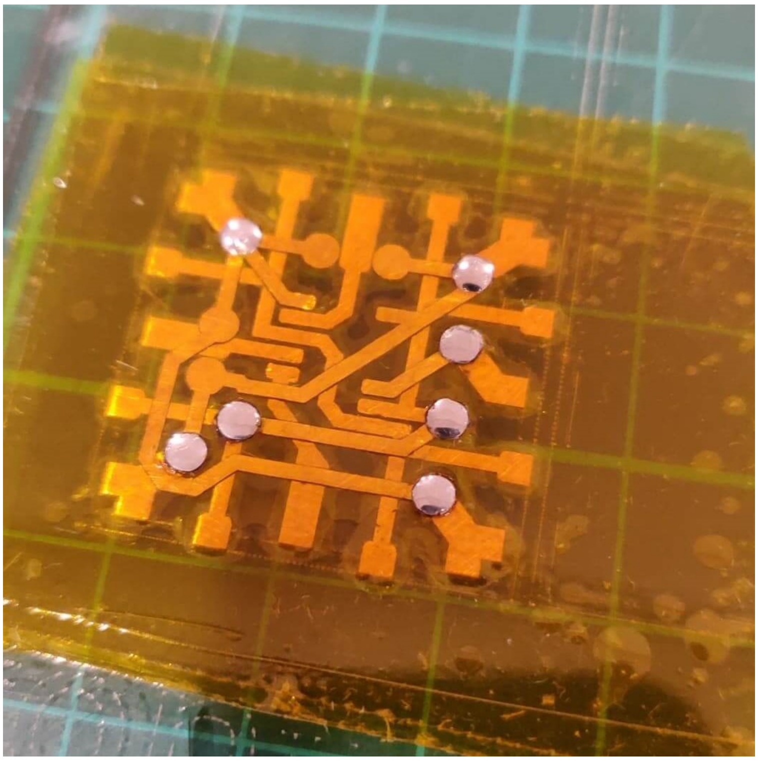


FIGURE 5.4: All VIAs between first and second circuit layer are soldered.

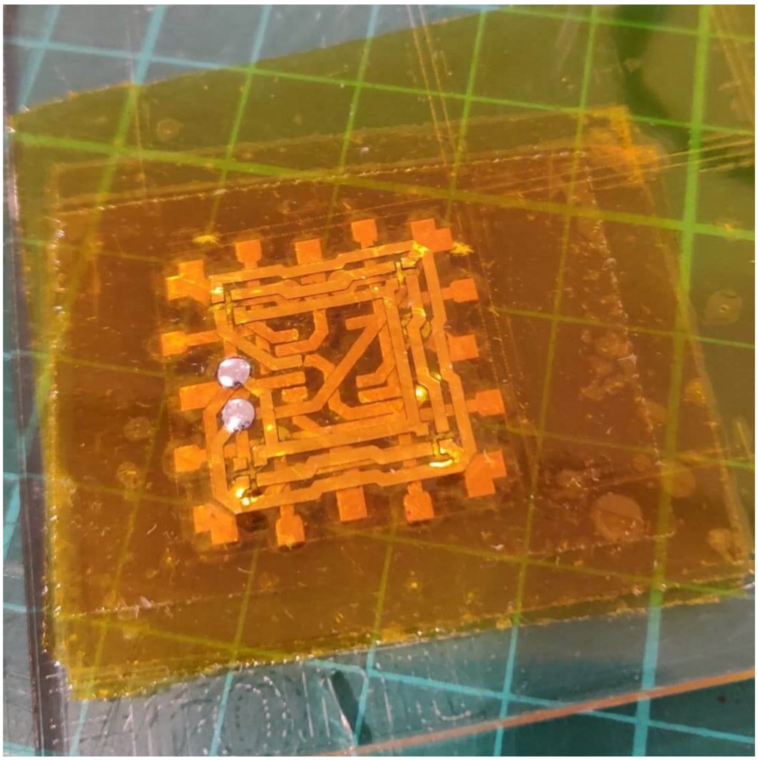


FIGURE 5.5: All VIAs between second and third circuit layer are soldered.

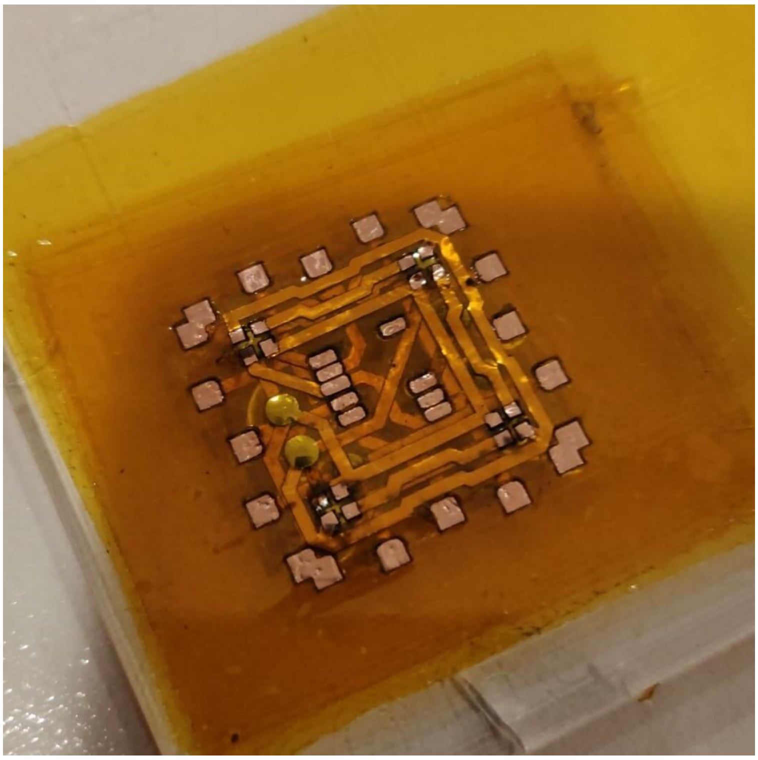


FIGURE 5.6: Connector and microcontroller pads cut out, the PCB is finished.

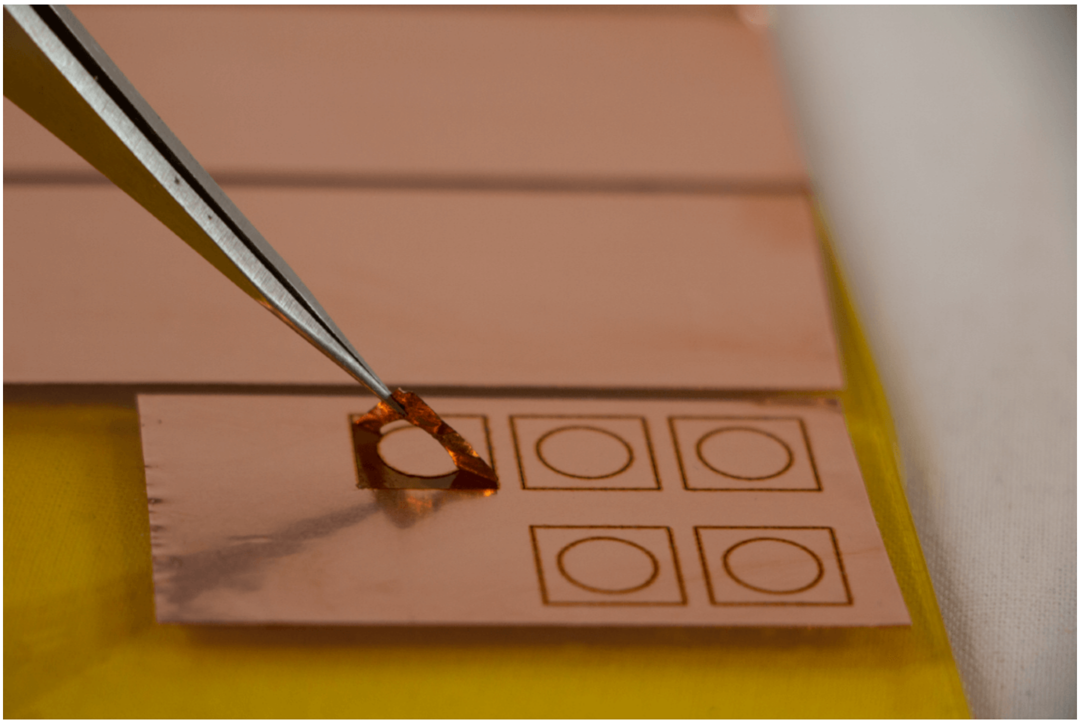


FIGURE 5.7: Calibration process of the laser.

FIGURE 5.8: Stress test of the PCB using a robot arm.

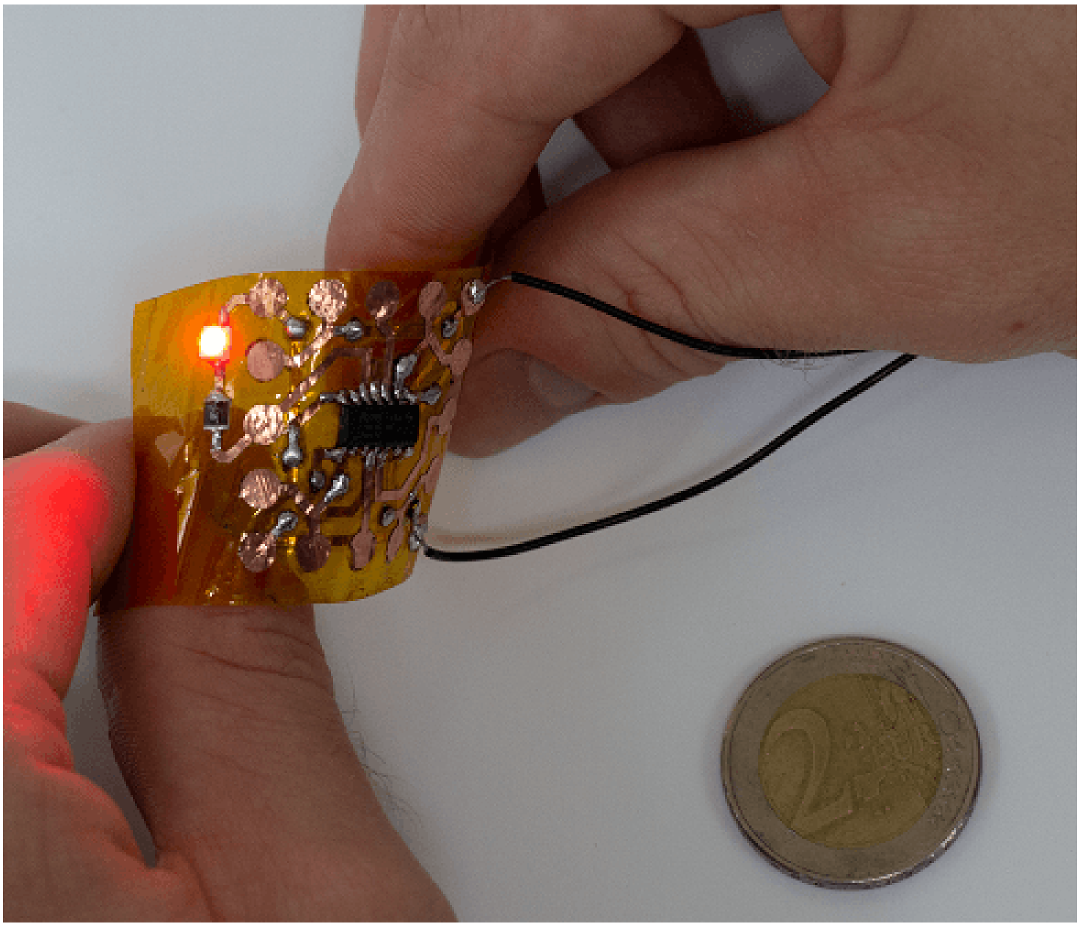


Figure 5.9: Full working flexible PCB programmed with a basic "blink" code.

# Chapter 6

# Architecture: Electronics, Protocols, and Algorithms

This chapter contains the architecture of the SoftMod modules by giving a detailed explanation of the decisions made when selecting the required electronics and protocols. The chapter starts with section 6.1, which describes the selecting procedure for all hardware components in the different modules. Next, section 6.2 explains and compares different communication protocols and decides which one is best suited for SoftMod modules. Section 6.3 describes all implemented algorithms, followed by section 6.4, which goes deeper into the implementation details. Finally, section 6.5 gives a brief overview of how to develop code on the chosen microcontroller.

## 6.1 Electronic Components

This section describes all used electronic components and hardware for each module. Since all modules embed the same microcontroller, subsection 6.1.1 dives deeper into the specifications of this component. Next, subsection 6.1.2 lists the additionally needed hardware for the master module, followed by subsection 6.1.3, which does the same for the slave modules.

### 6.1.1 Microcontroller

All SoftMod modules embed an ATtiny1614 [11] microcontroller operating at 10MHz. The ATtiny1614 is a low-cost (0.55 euros) and low-power (3.1 mA) microcontroller with a sufficient number of GPIO pins (12 pins) to control all modules' features. It has built-in hardware support for I2C, SPI and UART, 16 KB of programmable flash memory, 2 KB of SSRAM, and 256 bytes EEPROM. The ATtiny1614 can work on 3.3V, which is an advantage because most common sensors only accept 3.3V, and thus a voltage converter is not needed.

### 6.1.2 Master Module

The master module contains no additional sensing components, although some buttons and LEDs were added. First, a push-button enables reprogramming of the master by exposing the program pin of the ATtiny1614 to a contact in the connector. Next, a physical switch allows easy change of the behavior described in section 3.3. At last, an RGB LED shows the actual state of the master module by a series of blinking patterns. End-users can reprogram this LED to save energy or switch functionality.

### 6.1.3 Slave Modules

Almost all slave modules contain additional hardware that provides extra functionality. This hardware is described briefly by the following six paragraphs. The last paragraph also compares the power consumption of each slave module.

**Led and RGB Led Module**
The led module contains four regular green surface mount led's with fitting resistors (100 ohms). Each led is connected to a different pin, which makes the led's individual controllable. The RGB led module contains four WS2812C [75] LEDs that work with an input voltage of 3.7-5V. The RGB LEDs are daisy-chained[1] together what makes that there is only one microcontroller pin required to individual control the color of a led. Those type of RGB LEDs is also commonly used in RGB led strips.

**Temperature Module**
This slave module contains one analog temperature sensor with type MCP9700T [46] connected to an analog input pin of the ATtiny. Reading this value can be used to determine the temperature.

**Accelerometer Module**
This module contains a hard-PCB accelerometer module with type ADXL345 [2]. The pre-soldered PCB reduces the complexity and connects with the ATtiny through an SPI port.

**Button Module**
The only functionality of the button module is processing button presses; thus, the only component it contains is a default push-button. An internal pull-up resistor of the ATtiny is used to avoid floating pin states.

**Wireless Module**
Interaction with the outside world can happen with the wireless module. This module contains an ESP8266 [22] WiFi module that is connected with the ATtiny using a UART connection. The ESP8266 uses 3.3V as an input voltage.

**Battery Module**
The battery module is responsible for the power delivery to the circuits. It contains a one cell 3.7V 190mAh Li-Po battery [43] connected to an LDO voltage regulator of 3.3V [65]. This regulator reduces the voltage of the battery (healthy Li-Po's have a voltage range of 3V to 4.2V) to 3.3V. However, since the regulator cannot work with any voltages lower than 3.3V, the power output will be disabled when the battery reaches 3.3V. This inherent protects the battery from discharging below the minimal allowed discharge voltage of 3V. The input and output lines of the voltage regulator have a ceramic capacitor of $1\mu F$ [71] to create a stable voltage that can be used by all hardware components.

**Power Consumption**
Table 6.1 gives an overview of the power consumption of each slave device.

[1]Daisy-chaining means that the output of the previous component connects to the input of the next component.

| **Slave Module** | **Power Consumption (mA)** |
|---|---|
| **Temperature module** | 8.1 |
| **Accelerometer module** | 6.2 |
| **Push-button module** | 3.2 |
| **Led module** | 13 |
| **RGB led module** | 45 |
| **Wireless module** | 85 |
| **Cable and adapter modules** | 3.1 |

TABLE 6.1: Comparison of the required power to run each slave module.

## 6.2 Communication and Protocol Stack

Since the SoftMod master module needs to control slave modules and be able to communicate with a computer, two communication protocols must be selected. Protocols supported by the hardware of the microcontroller, in this case an ATtiny1614 (see subsection 6.1.1), can reduce the complexity of the implementation. Because SoftMod uses soft and relatively unstable connectors (described in section 4.1), it is better to implement a less complicated protocol to avoid inaccuracies and thus reduce data-loss. Unstable connectors can increase the noise level on the communication lines since there is no direct permanent interconnection between two microcontrollers. This instability can cause problems with high-speed protocols since a slight increase of resistance or even a detachment of as little as one nanosecond can interrupt the data flow and thus create invalid data. Besides, some protocols are only designed to work for a maximum distance of a few centimeters on a PCB and are not suited for SoftMod.

The next subsection 6.2.1 describes the requirements for the communication protocol that the master module uses to communicate with the slave modules. Subsection 6.2.2 does the same for the interaction between an assembly and computer. After that, subsection 6.2 compares a pre-selected list with possible protocols and gives a brief overview of the essential workings. Finally, subsection 6.2.4 concludes this section by selecting the most beneficial protocols to use as inter-module and computer-module communication within SoftMod.

### 6.2.1 Inter-Module Communication

Inter-module communication always requires that the master can control every other slave device connected to the assembly. The protocol should in order to meet this qualification, be bidirectional, and provide some bus, daisy chain, or mesh network capability. Bus protocols have relatively fast data speeds and are easy to implement, but require some addressing system in order to address a specific device. Daisy-chaining devices, thus linking the output of one device with the input of the next device, require more sophisticated algorithms. It also slows down the transmission rate of the network because every data packet has to be processed by every device in between source and destination. Mesh networking with routing algorithms is very difficult and complex to implement but is generally speaking faster than a daisy-chained network.

On a bus communication, there can be one or more master devices. In a single master environment, there is only one master device that can initiate a data transfer. However, multi-master environments have multiple masters that can initiate a data transfer. They are thus more complex tom implement and require arbitration and collision checks.

Communications can be half or full-duplex. Half-duplex means there can be a data transfer only in one direction, while a full-duplex communication protocol can transmit data in both directions at the same time.

### 6.2.2 Module-Computer Communication

SoftMod only contains one module (the master) that has to be able to connect to the programmer and computer. The module-computer protocol should be simple and easy to implement and provide a bidirectional communication that also can be understood by a computer without too many additional components.

### 6.2.3 Protocol Comparisons

A comparison has to be made to find the protocol that meets most of the above requirements. Possible protocols are I2C, SPI, 1-Wire, and UART. The following four paragraphs give a brief overview of each protocol and discuss the pros and cons of their use in SoftMod. The second last paragraph describes less common protocols, followed by the last paragraph, which shows the results of a practical speed test.

**I2C**
I2C is a half-duplex bidirectional bus communication protocol that uses only two wires: SCL (clock) and SDA (data). I2C is also fully supported in the hardware of the ATtiny what makes implementing it straight forward. I2C works with acknowledgments after every sent byte, so it is a robust protocol. I2C can work in both single and multi-master environments.

To address different devices connected to the bus, I2C uses a 7-bit address matching system. Data transmissions are always initiated from a master and start by sending the 7-bit address byte and a bit that indicated if the master wants to read or write. The slave with the corresponding address acknowledges the address match upon which the master continues with the data transmission. Since there are only 7-bit addresses, there can only be 127 devices connected to the same bus. However, implementing a full software address matching algorithm can extend this number to an unlimited number of bits for the address. The ATtiny supports besides the default 7-bit address also a 10-bit address, but the hardware can only match the seven most significant bits. The software is responsible for handling the other three.

Theoretically, I2C can achieve speeds of 100 kbit/s and 400 kbit/s full speed in standard mode. In fast mode, I2C has ratings of 1 Mbit/s and 3.2 Mbit/s full speed. [33]. I2C can change the transmission rates dynamically based upon processing time and the clock speeds of the microcontrollers.

**SPI**
SPI can achieve speeds of up to 10 Mbit/s [21] and uses 3 wires: CLK (clock), MOSI (master-out slave-in data) and MISO (master-in slave-out data), and one additional wire per connected module: SS (slave select). Implementing a daisy-chain or mesh

structure omits this wire but increases complexity. In that case SPI also requires an addressing system similar to that of I2C.

SPI is a full-duplex bidirectional protocol that works by just streaming the data. There are thus no parity checks nor acknowledgments nor error checks which make the protocol less robust but easier to implement. The hardware of the ATtiny also supports the SPI protocol.

**1-Wire**

1-Wire is a half-duplex bidirectional device communications bus system that provides low-speed data, signaling, and power over a single conductor [1]. 1-wire can be implemented in software as well in hardware. The hardware of the ATtiny does not support 1-Wire, so additional components are required. The DS2480B [20] UART to 1-Wire Line Driver can be used as a master transmitter since the hardware does support UART. 1-Wire can also be fully implemented in software, and since the timings of 1-Wire are not that fast, this is not too complex. Both sender and receiver must implement the same timing schedule and there can be no interruptions while transmitting data.

Powering all devices through 1-Wire is a nice benefit, but requires more additional components like capacitors to fill the gaps caused by zeros in data. Also, powering high-current components like, for example, multiple LED modules can cause stress on the single line.

Several tests with software and the additional hardware concluded that the protocol works very well, but it is not recommended to use within SoftMod because of the low transmission rates. 1-Wire can work with high-speed data transfers, but the reliability is very low due to the unstable connectors.

**UART**

UART, or better known as Serial, is a full-duplex bidirectional communication protocol that uses two wires [12]: TX (outgoing data) and RX (incoming data). The user predetermines and hardcodes the data transmission speed in the sender and receiver, so UART does not require a clock line. Serial is relatively slow in comparison to I2C and SPI, with a baud rate of 9600 the transmission rate amounts 9.6 kbit/s. When using a TTL to USB adapter, UART can be used to communicate with any USB device, like a smartphone or computer. The hardware of an Arduino Uno can even function as a converter by connecting the reset pin to ground.

UART data packets contain one start bit, one stop bit and one parity that provides some error detection, which makes UART a quite robust protocol.

**Less Common Protocols**

Some less known protocols can also work as communication protocol, such as the protocol used to control WS2812 LED strips [42]. This protocol works over one wire and is designed to be very fast, but only works in one direction. Since the signal combines clock and data into one, this protocol is more error-prone than 1-Wire, but it requires stringent timings. The protocol can combine clock and data into one signal by making use of edge[2] detection. If the time before an edge is longer than the time, a zero is transmitted, else a one.

[2] An edge is when the line goes from low to high or high to low.

**Speed test**

Table 6.2 shows a comparison of the effective transmission rates achieved by most important considered protocols. All values represent the time in milliseconds it takes to transmit one byte of data to the slave microcontroller, and each test is an average of 10 independent tests. An Arduino Uno represented the master microcontroller; the slave was an ATtiny1614. I2C, 1-Wire, and UART were implemented using a 7-bit addressing system. For the SPI protocol, there were no algorithms implemented to support multiple devices without a separate wire for each device. UART was implemented using a baud rate of 9600.

| Protocol | Required number of pins | Time to send 1 byte of data (ms) | Time to send 10 bytes of data (ms) |
|---|---|---|---|
| **I2C** | 2 | 0.16 | 0.88 |
| **SPI** | 4 | 0.08 | 0.79 |
| **1-Wire** | 1 | 0.98 | 5.40 |
| **UART** | 2 | 0.21 | 1.15 |

TABLE 6.2: Comparison of the required number of pins and transmission rates for each protocol candidate.

### 6.2.4 Conclusion

For inter-module communications, I2C is chosen over the alternatives, as it requires fewer wires and has superior transmission rates. I2C is supported in hardware by the ATtiny and is straight forward to implement. I2C is by default a bus communication protocol so no further algorithms should be implemented. The only downside is the limited amount of possible addresses, but an address resolution system based on hardware collision detection fixes this problem (see subsection 6.3.1).

UART is selected for the computer-module communication protocol since it can be used directly with a computer when using a UART bridge. UART is also supported in hardware by the ATtiny.

## 6.3 Smart Algorithms

Unlike LittleBits [13] and MakerWear [36], the behavior of a module is not hard-coded. A module can be controlled by the master module based on the physical configuration of modules, or by user-specified behavior in the master module (section 3.3). Therefore the master module needs to be able to control, and thus address (subsection 6.3.1), slave modules and track their topology (subsection 6.3.2). The user can also see this topology when the master module connects to a computer (subsection 6.3.3). Finally subsection 6.3.4 describes the overall algorithm the modules uses to execute all tasks.

### 6.3.1 Addressing Slave Modules

The I2C standard uses 7-bit addressing space and thus supports unicast addressing of a maximum of 127 modules on a single bus. While this would be enough for making advanced prototypes with SoftMod, hard-coding the I2C address in every module would result in duplicate addresses when combining different SoftMod sets. One solution would be to broadcast all messages and add the ATtiny's 64-bit unique identifier as data. However, as I2C transmits data byte per byte, sending one byte of data would first require transmitting an 8-byte address, which would give a significant overhead. Therefore, the master module incorporates a dynamic address allocation technique inspired by the ARP protocol used by SMBus. This technique, described below, dynamically assigns a 7-bit I2C address to slave modules connected to the master module (directly or through another slave module). The master module ensures that the 7-bit address for every slave module is unique within a SoftMod assembly.

When powered, slave modules have by default I2C address 127 (ARP address). The master module continuously sends a read request to this ARP address and immediately receives a response from new slave modules. Before the master module can assign unique I2C addresses to the new slave modules, it needs to address every new slave module individually. Therefore, the master module sends out a request to initiate a self-selection strategy on the ARP address. New slave modules that receive this request, all start transmitting their ATtiny 64-bit unique identifier over I2C. Physically this is done by pulling the data line (SDA) low or releasing it every clock cycle (SCL) as shown in Figure 6.1. Releasing the SDA line makes it high since the I2C lines are high by default. When a slave module detects a collision, it must immediately stop communicating its identifier. Collisions can be detected when the slave module is reading that the data line is low after releasing it in the same clock cycle. After 64 clock cycles, only one new slave module could transmit the entire unique identifier and accepts the unique 7-bit I2C address that the master hands-out by making a read request to a new I2C address right after 64 cock cycles. Finally, the slave module replies to the master module and communicates its module type. This strategy continues until no slaves respond to the ARP address.

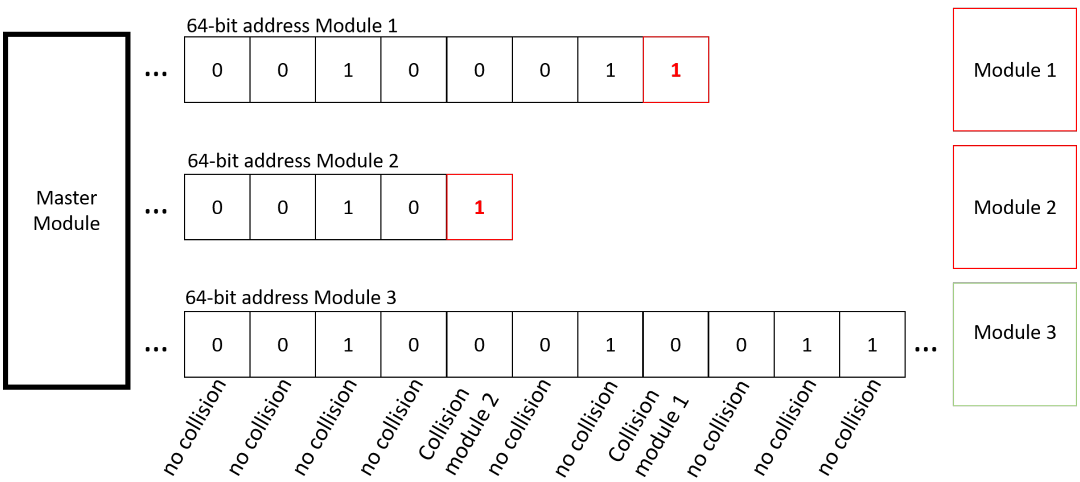



FIGURE 6.1: SoftMod uses a dynamic address allocation technique to assign a 7-bit I2C address to all modules. During this procedure, a single slave module is selected by having all modules transmit their 64-bit ATtiny address over the I2C bus while listening for collisions.

Four I2C addresses are reserved: broadcast, master module address (used for the topology tracking), I2C to UART module (used for the topology visualization) and ARP). Slave modules can thus make use of the 124 other addresses that are still available.

### 6.3.2 Topology Tracking

When all new slave modules have a unique I2C address, SoftMod can update the topology of the assembly. Therefore, the master module transmits four read requests to each new slave module to retrieve the addresses and the orientation of the four connected neighboring modules. For one module to request the address of a neighbor module, it cannot use the I2C lines as this bus reaches every other device. Thus every request is transmitted to all modules in the assembly, which is unwanted. Therefore a fifth electrode, the trigger, is present in the magnetic connector. This electrode directly connects a digital pin of one module's microcontroller to a digital pin of a neighboring module's microcontroller. As there are four edges on a module, four digital pins on every microcontroller are reserved for trigger electrodes.

Neighboring modules transmit their 7-bit I2C address, plus an additional two bits to identify the connector (orientation of the module), over this trigger electrode by pulling it high or low for all 9 bits. In contrast to I2C, no clock is present, and the transmission is strictly time-based. To correctly read and write all bits, both modules go into an atomic state during which they cannot be interrupted by other requests. Figure 6.2 illustrates this process. (1) When module A needs the address and orientation of a neighbor module B, module A generates an interrupt at module B by pulling the trigger from high to low for 5ms after which module A goes into the atomic state and is ready to read bits. (2) From the moment module B is idle and detects this interrupt, it goes into the atomic state and transmits a start signal by pulling the trigger low for 5ms. When no start signal is received after a 100ms timeout at module A, the process stops, and module A reports to the master module that no neighbor is attached at one edge. (3) When both modules are in the atomic state, module B transmits the 9-bits by pulling the trigger low or high every 5ms while module A reads at the same interval. (4) After the transmission is finished, module A transmits all 9-bits to the master module over I2C, which in turn updates the topology accordingly. This protocol is based on the 1-Wire protocol.

### 6.3.3 Topology Visualization

The SoftMod IDE visualizes the topology when the master module connects to a desktop computer. This application is implemented for Windows in C#. A custom-designed I2C to UART bridge transfers the data from the master module to the computer. The microcontroller of the bridge is also an ATtiny1614 and only buffers and converts I2C data to UART data. The bridge, however, has a unique reserved I2C address so the master module can detect its presence on the bus. As soon as the master discovers this address, it starts transmitting all the topology data. Whenever the topology updates (this is when a new device connects, or an old one disconnects) the master resends all topology data.

### 6.3.4 Controlling the Assembly

Assigning a unique I2C address to all new slave modules and updating the topology takes a few 10 milliseconds depending on the number of new modules and the complexity of the topology. After this automated initialization process, the master modules proceeds executing the default or user-specified behavior. During this process, it continuously reads updates from input modules and updates the state of output modules whenever needed.

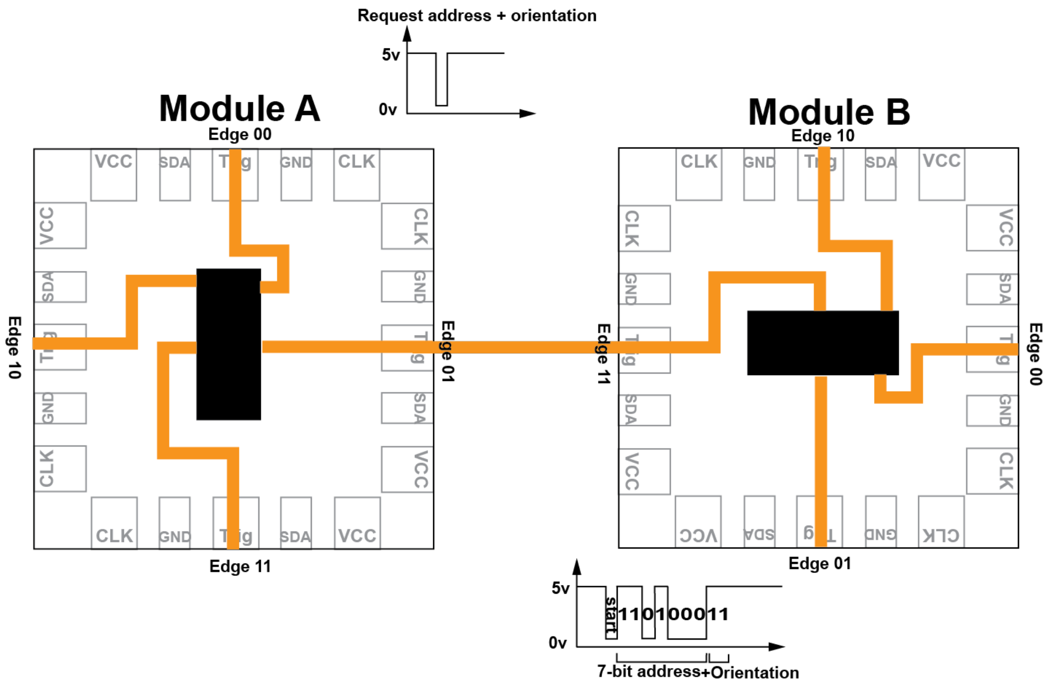



FIGURE 6.2: Slave modules identify neighbors and their orientation by transmitting the 7-bit address and connector code over the trigger electrode.

## 6.4 Module Implementations

This section dives deeper into the most important implementation aspects of the modules. First, subsection 6.4.1 describes the code that runs on a master module. Next, subsection 6.4.2 explains the implementation details of the slave modules. Subsection 6.4.3 goes over the smart implementation of the I2C protocol without using buffers. Finally subsection 6.4.4 ends this section by describing the use of persistent memory to store the unique identifiers.

Since the ATtiny1614 is relatively new, there are no existing libraries available the code must implement direct hardware register access. Coding with registers can be quite hard but gives the full control over all functionalities of the microcontroller.

### 6.4.1 Master Module

The code of the master module contains one big loop that repeats itself every ten milliseconds. First, the master checks for new devices by sending the ARP address. If there are new devices, the master updates the topology by questioning the new device. Next, the master checks if the I2C to UART bridge is connected and dependent on if there where updates in the topology the master sends all data. After that, the master questions all input devices for new data in order to calculate and send the new states of the output devices. If an input or output device does not respond, the master assumes that the device is disconnected and removes it from the topology. After this, the master loop repeats itself.

**Status RGB LED** The master module is the only module that contains an RGB status LED, which shows the current state of the device. When the master module gets powered, the led flashes blue or green to show the selected behavior, respectively default, or user-specified (see section 3.3). By using the physical switch the behavior of the master module can be changed. Changing the behavior, however, has to be done before the device powers up. After that, the led flashes once when it receives its I2C address. From there on the led blinks at the start of each master loop iteration. A precise timer implementation allows the led to flash for a specific amount of time.

### 6.4.2 Slave Modules

The slave code does not contain any loops and is purely interrupt based. Actions are executed by reading or writing to fictional registers. The master does this by first writing the address of the register to the slave module in order to set this register as active. Next, the master executes the wanted read or write operation to execute the action corresponding with the set active register. For example, if the master wants to read a button state, it first sends the address of the register that corresponds with the action "get last button state" followed by a read request that reads this data. Setting the color of an LED requires two write operations, first the register address of the color (red, blue or green) that is going to be set, next the intensity of this color.

Some registers are also linked, so the master does not have to write each register address. The topology registers are a perfect example of this functionality. When the master reads the register that contains the address of the device connected to the top connector, the slave automatically updates the active register to the register that contains the address of the bottom device. Because of this procedure, the master can read an array of data which significantly reduces the overhead.

In code, callback methods are used to simulate all registers, which has the benefit that they always contain the most up-to-date data. The downside of this approach is that the I2C communication has to wait for the callback method to finish. This waiting, however, is no problem for the I2C protocol since it supports waiting for data operations.

### 6.4.3 Bufferless I2C

Since all modules use an ATtiny1614 there is only 2KB SSRAM available. This amount of memory is not much since all incoming and outgoing I2C data needs to be buffered, and the master module also needs to store the full topology. As an example, the official Arduino I2C implementation [67] uses five buffers of 32 bytes to handle all I2C communication. The I2C implementation on the ATtiny has been made bufferless by using callbacks and interrupts. This method assures that every byte is stored at most once and thus reduces memory usage.

### 6.4.4 Persistent Memory

Each ATtiny1614 has 256 bytes EEPROM which is a persistent memory that not gets erased on upload or power loss. This memory is convenient as storage for the 64-bit unique identifier of the module. With this technique, the modules can be reprogrammed without changing the identifier for every upload to another device.

## 6.5 Developing Software on the ATtiny1614

Programming and debugging an ATtiny1614 can be quite hard without the proper tools. Subsection 6.5.1 describes the programming method of the ATtiny1614. The

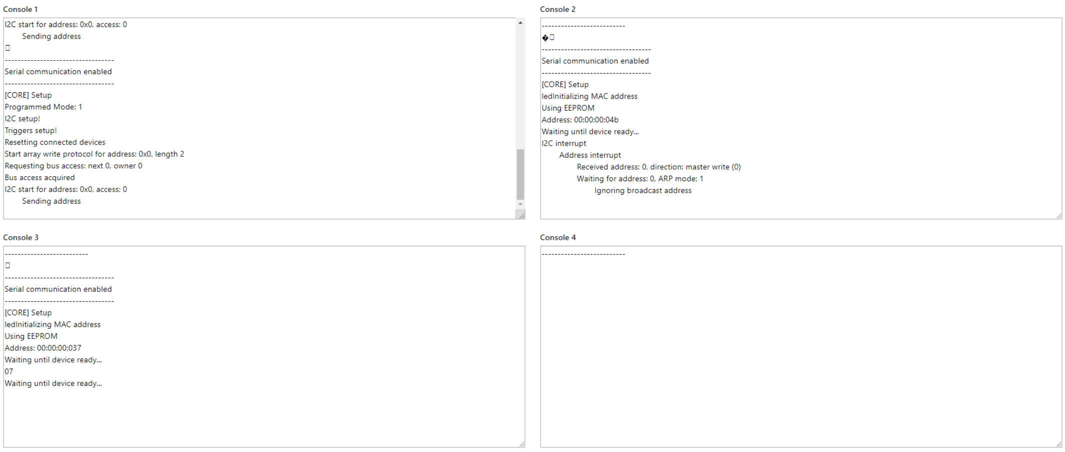



FIGURE 6.3: Multi-console window showing *printf* statements sent over UART of up to four separate ATtiny modules. Console 1 displays the messages of the master module, Console 2 and 3 the states of two slave modules.

following subsection 6.5.2 explains the implemented debugging methods like four parallel serial consoles. The last subsection 6.5.3 shows the full development setup like custom-built debugging shields that allow easy programming and debugging of up to four devices at the same time.

### 6.5.1 Programming

An Arduino Uno can be used as a simple programmer, but it is more recommended to use an AVR programmer like the Atmel ICE [9] in combination with Atmel Studio [8]. This programmer provides full debugging capabilities with breakpoints, step-through functionality, and the monitoring of variables.

### 6.5.2 Debugging

Besides debugging with the Atmel ICE, it is also useful to print some statements to the console. Implementing a UART communication that overrides the default *printf* functionality provides this extra debugging option. The setup mentioned above allows the use of *printf* and its formatting as usual, but instead of the standard output stream, UART sends all characters to an Arduino Mega [6]. This Arduino combines UART streams of up to four ATtiny's by assigning an identifier to each stream of bytes and sends this total stream to a connected computer. This computer runs a NodeJS [23] server which serves a web page with four console windows that display all incoming text, as shown in figure 6.3. The system thus provides *printf* output of up to four modules at the same time. A custom-created batch script simplifies the setup like COM port selection and server initializations.

### 6.5.3 Custom Created Hardware

Programming the ATtiny while plugged in a breadboard is fine, but when working with multiple microprocessors and inter-communications, it saves time to create and use a custom-developed shield with all necessary connections, like power, programming headers, I2C (including pull-up resistors) and UART debugging already soldered. The

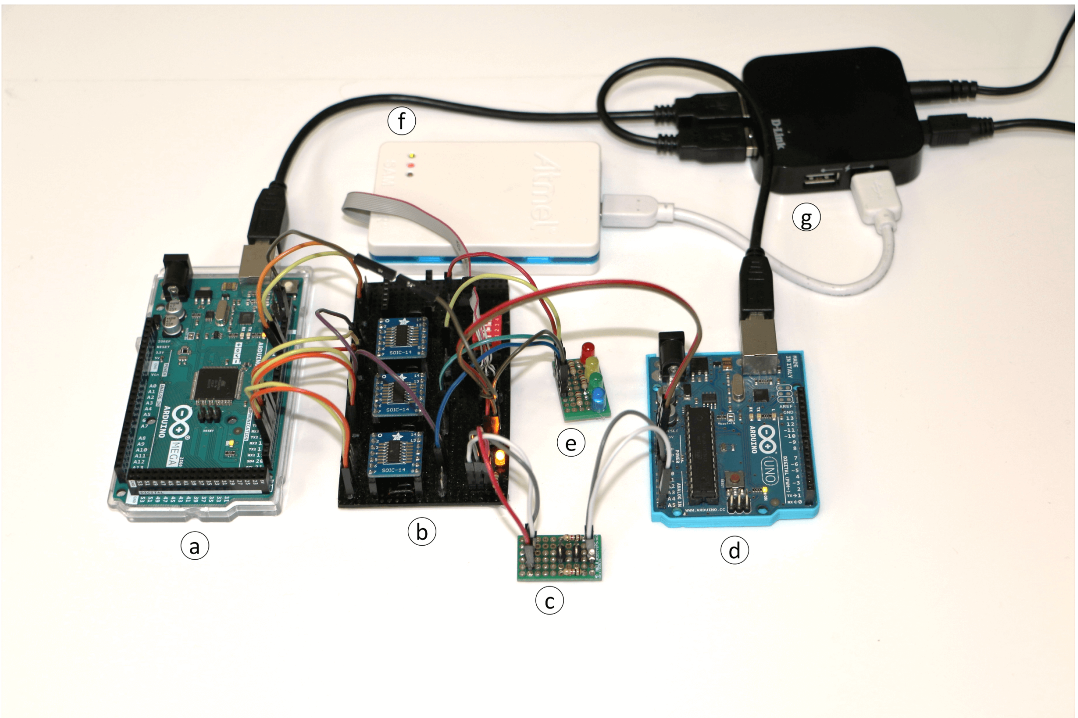



FIGURE 6.4: Custom-developed shield used to program and debug up to four ATtiny1614 devices, containing a) Arduino Mega as UART debugger, b) custom created programming shield, c) I2C pull-up bridge, d) Arduino Uno as temporary master module, e) shield to debug with LEDs, f) Atmel ICE programmer and g) external powered USB hub connected to a computer.

shield also contains a DIP switch to select which module is going to be programmed by redirecting the programming line. Figure 6.4 shows this full programming setup.

# Chapter 7

# From SoftMod Assembly to High-Fidelity Prototypes

SoftMod is not a toolkit that users can only use to prototype. As mentioned in this thesis, SoftMod assemblies can also be converted to high-fidelity prototypes, or transformed into rigid shapes to be used for evolutionary prototyping. There are a lot of different techniques available to do this. For example, casting resin or silicone over SoftMod assemblies will reinforce them. Module assemblies can also be fully embedded inside casted objects as shown in the boat in figure 7.1. A SoftMod assembly could also be used to enhance 3D objects and to provide additional functionality. Another technique is to create a whole new solid circuit based on the characteristics of the created module assembly. This new circuit can be made out of one piece and thus does not need any connectors. The circuit could even be optimized by using one microcontroller that providing the same functionality of all modules.

This chapter describes a technique proposed by SoftMod to determine the 2D position of components given their 3D position on an object. In other words, this technique allows to visually place components on a 3D model after which a 2D module assembly can be generated that, once placed on top of the 3D model, contains the placed modules in precisely the right position and orientation.

Following sections explain how this technique works. First, section 7.1 gives a brief overview of how this technique works. Next, section 7.2 describes the specifications of the calibration pattern. Finally, section 7.3 describes the software tool that is developed in order to do the conversions.

## 7.1 Concept

To be able to convert virtual 3D module assemblies to 2D, the object needs to be processed. First, a calibration pattern has to be applied to the object to track the curves and shapes of this object. This pattern can also be stretched, for example when used with thermoforming. The software tool will automatically take the stretchability into account when calculating the 2D points. Next, a third-party scanning tool called AgiSoft [3] creates a 3D shape of the physical object by using just pictures. This 3D model is then remeshed to improve the speed of the algorithm used to convert the assembly. The software tool visualizes the scanned object and allows users to create a virtual SoftMod assembly.

## 7.2 Calibration Pattern

The calibration pattern consists out of different colored squares, as shown in figure 7.2. The seven colors on the top-left represent all possible colors and serve as

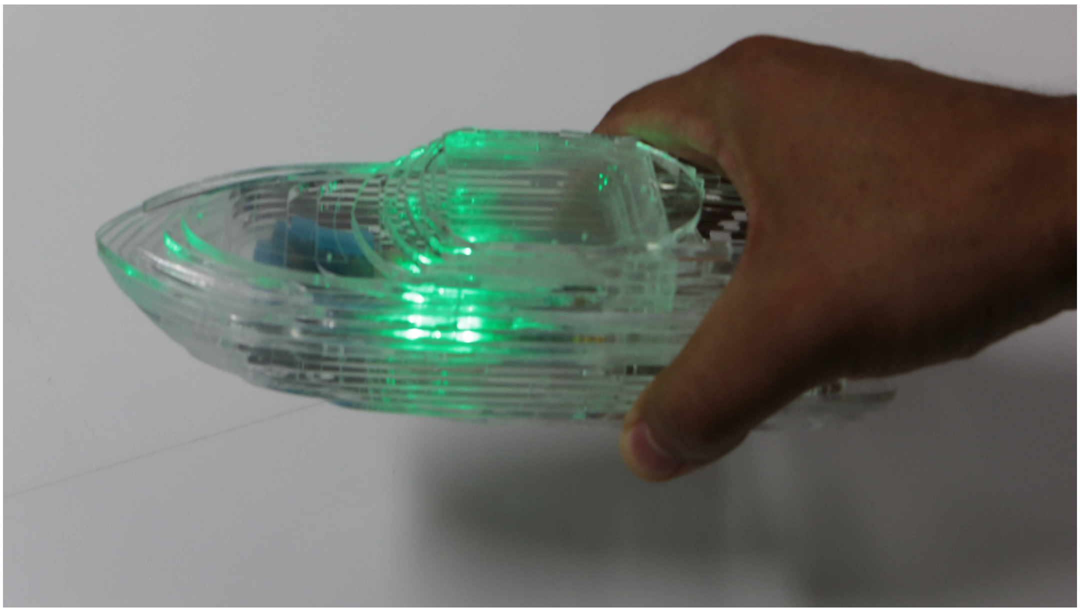


FIGURE 7.1: A SoftMod assembly embedded in a boat made out of plexiglass. The led turns on or of when the boat is shaken.

a color calibration pattern for each color. The unique property to this grid is that each combination of 3x3 colors can occur only once in the entire grid. This calibration grid works as a coordinate system since each 3x3 color grid detected on in 3D space corresponds with exactly one color grid in 2D space. This mapping provides an exact location in 2D, even when the calibration pattern gets stretched. With the interpolation of the 3x3 color grid, an exact point can be determined.

## 7.3 Software Tool

The software interface of the tool, as seen in figure 7.3, loads a scanned 3D model and configures all settings by an automated setup wizard. Each color in the scanned model is also converted back to it's original color, because colors can change during printing and scanning. Next, the user can place modules on the 3D model, just like the plug-and-play approach provided by the physical SoftMod modules. When the user is finished the created design gets converted to 2D and the software tool shows this layout.

The algorithm behind the conversion makes use of a custom generated point cloud. Because the model is remeshed and thus triangles are almost identical, vertices are used as points in this point cloud. When a 3D point needs to be converted, the algorithm has to find the 3x3 color grid surrounding the selected point. First, the algorithm will look for the closest point in the point cloud and sets this as the start point. Next, the algorithm rotates and transforms the model so the start point is at (0,0,0) and the normal in that point is facing up. The algorithm then retrieves all point cloud points in a predetermined radius around the center point and projects them on a horizontal plane. Those transformations remove one dimension and improve the accuracy of the detection. Next, the points in the plane are divided into a predetermined set of circle slices. Each circle slice counts the number of points that have the same color as the center point, to determine the color of the center square. This counting starts with the point closest to the center and ending with the farthest

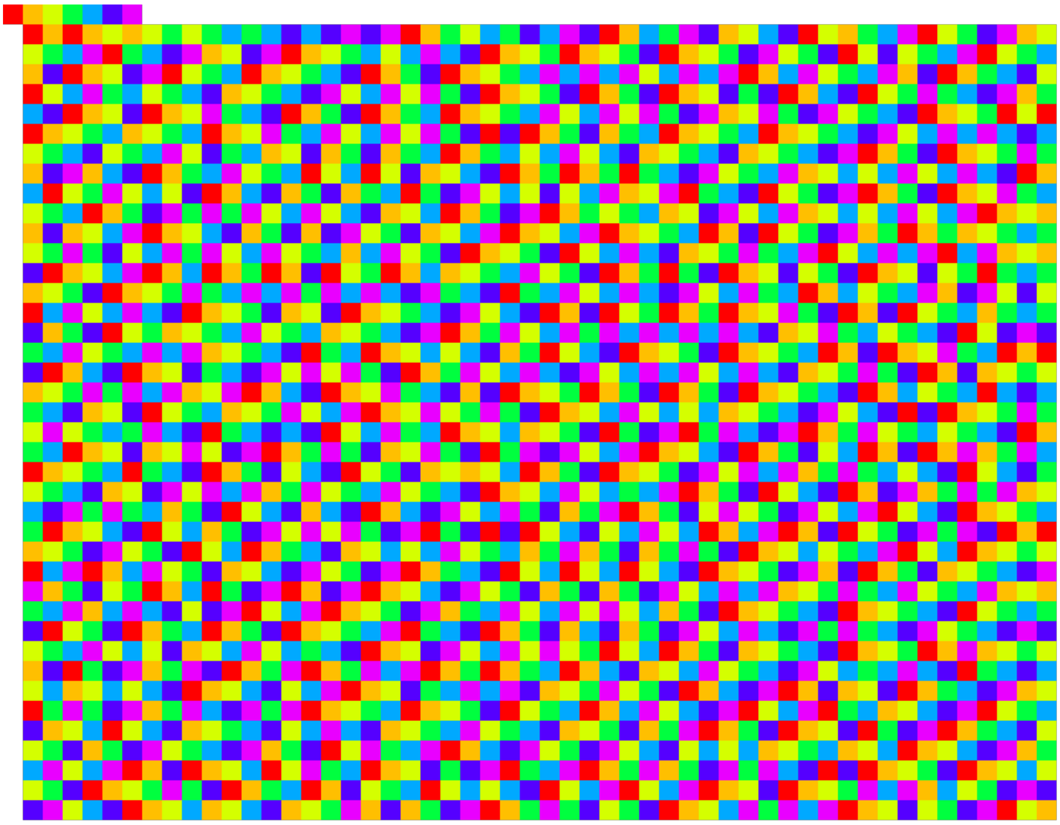


FIGURE 7.2: The calibration pattern used to cover objects.

one. If the count is below a predetermined threshold, a penalty will be added, and a next color is selected. After this process, all weighted sums determine the color. For the surrounding squares, the same technique is applied, but now the number of points is counted starting and ending in a predetermined range.


Figure 7.3: Software tool to convert stretched 3D coordinates to 2D using a calibration pattern.

# Chapter 8

# Limitations, Future Work and Conclusion

This chapter finalizes the thesis by giving an objective overview of the limitations and future work of SoftMod in section 8.1. Section 8.2 goes deeper into the limitation SoftMod has as a new prototype technique. Finally, section 8.3 concludes this thesis.

## 8.1 Technical Limitations SoftMod

SoftMod has four technical limitations, which are going to be addressed in future versions of the system:

First, while the SoftMod IDE supports specifying basic If-This-Then-That (IFTTT) behavior, more advanced logic requires composing code in scratch-like programming environments supporting Arduino (i.e. S4A) or the Arduino IDE. These programming paradigms are still quite challenging for novices, especially for distributed systems communicating over WiFi. Therefore, future versions of SoftMod could support techniques that further lower the barrier for end-users to specify sensor behavior, such as IBM NodeRed [54], Pulsation [59] or Trigger-Action-Circuits [4].

Second, a modular kit can always be extended and upgraded with more modules. For example, cable modules can be extended with deflection sensing to offer new modalities for input and further increase the accuracy of the 3D assembly reconstruction in SoftMod's IDE. The wireless communication module can become flexible when thin-film flexible antennas become available. The power module could embed a capacitive pad and three LEDs for users to check the battery status and recharge them in time. SoftMod could also be extended with entirely new modules, such as modules for actuation and advanced sensing.

Third, there could still be an improvement in the connector design. Currently, the connectors do not prevent short-circuits because of a mistake in the last iterations. Suggested is that a few pins switch place in order to prevent that ground and 3.3V gets connected directly. All other incorrect connections make the assembly to fail, but this can be detected and reported by the software. Even when users would enforce such inappropriate connections, the connectors are designed to prevent short circuits.

Last, all modules embed an ATtiny1614 microcontroller for processing. Although this is sufficient for all slave modules as they only run the software framework for communication, the master module stores the topology, the default behavior, as well as the user-defined behavior. This could exceed the 2kb of SRAM of the ATtiny1614. In future versions, we plan to use a more powerful microcontroller for the master module, such as the Microchip SAM L10[10] that embeds a Cortex-M23 CPU and has 16kb of SRAM. This amount of RAM is especially useful for more advanced end-user specified behavior as these programs are often less efficient in terms of memory consumption.

## 8.2 SoftMod to High Fidelity Prototypes Limitations

Evolutionary prototyping,....

## 8.3 Conclusion

In this thesis, we presented SoftMod, a modular electronics kit consisting of soft and flexible modules that snap together for prototyping novel interactive systems. SoftMod offers qualities for both children and adults as it comes with simple plug-and-play behavior as well as advanced user-specified behavior. In this thesis, we contributed and detailed our software framework, electronic design, mechanical connector design, and prototyping procedures for making SoftMod modules. We hope our toolkit empowers designers, researchers, and artists, to explore new ideas for electronic systems, including wearables, interactive skin, and textiles. As the process to make our soft modules is entirely DIY compatible, we also hope the community builds further on our ideas and joints our efforts to build easy-to-use electronic toolkits.

# Bibliography


[1] *1-Wire - Maxim*. URL: https://www.maximintegrated.com/en/products/interface/one-wire.html (visited on 08/22/2019).

[2] *ADXL345 Datasheet and Product Info | Analog Devices*. URL: https://www.analog.com/en/products/adxl345.html (visited on 08/22/2019).

[3] *Agisoft Metashape*. URL: https://www.agisoft.com/ (visited on 08/28/2019).

[4] Fraser Anderson, Tovi Grossman, and George Fitzmaurice. “Trigger-Action-Circuits: Leveraging Generative Design to Enable Novices to Design and Build Circuitry”. In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. UIST ’17. event-place: Québec City, QC, Canada. New York, NY, USA: ACM, 2017, pp. 331–342. ISBN: 978-1-4503-4981-9. DOI: 10.1145/3126594.3126637. URL: http://doi.acm.org/10.1145/3126594.3126637.

[5] *Arduino - Home*. URL: https://www.arduino.cc/ (visited on 08/22/2019).

[6] *Arduino Mega 2560 Rev3*. URL: https://store.arduino.cc/mega-2560-r3 (visited on 08/22/2019).

[7] *Arduino Uno Rev3*. URL: https://store.arduino.cc/arduino-uno-rev3 (visited on 08/22/2019).

[8] *Atmel Studio 7 | Microchip Technology*. URL: https://www.microchip.com/mplab/avr-support/atmel-studio-7 (visited on 08/22/2019).

[9] *Atmel-ICE*. URL: https://www.microchip.com/DevelopmentTools/ProductDetails/ATATMEL-ICE (visited on 08/22/2019).

[10] *ATSAML10D16A - 32-bit Microcontrollers*. URL: https://www.microchip.com/wwwproducts/en/ATSAML10D16A (visited on 08/22/2019).

[11] *ATTINY1614 - 8-bit Microcontrollers*. URL: https://www.microchip.com/wwwproducts/en/ATtiny1614 (visited on 08/22/2019).

[12] *Basics of UART Explained - Communication Protocol, Block Diagram, Interfacing*. URL: https://www.elprocus.com/basics-of-uart-communication-block-diagram-applications/ (visited on 08/22/2019).

[13] Ayah Bdeir. “Electronics As Material: LittleBits”. In: *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*. TEI ’09. event-place: Cambridge, United Kingdom. New York, NY, USA: ACM, 2009, pp. 397–400. ISBN: 978-1-60558-493-5. DOI: 10.1145/1517664.1517743. URL: http://doi.acm.org/10.1145/1517664.1517743.

[14] Leah Buechley and Michael Eisenberg. “The LilyPad Arduino: Toward Wearable Engineering for Everyone”. In: *IEEE Pervasive Computing* 7.2 (Apr. 2008), pp. 12–15. ISSN: 1536-1268. DOI: 10.1109/MPRV.2008.38. URL: http://ieeexplore.ieee.org/document/4487082/ (visited on 08/22/2019).

[15] L. Buechley et al. "Quilt Snaps: A Fabric Based Computational Construction Kit". In: *IEEE International Workshop on Wireless and Mobile Technologies in Education (WMTE'05).* IEEE International Workshop on Wireless and Mobile Technologies in Education (WMTE'05). Tokushima, Japan: IEEE, 2005, pp. 219–221. ISBN: 978-0-7695-2385-9. DOI: 10.1109/WMTE.2005.55. URL: http://ieeexplore.ieee.org/document/1579269/.

[16] Varun Perumal C and Daniel Wigdor. "Printem: Instant Printed Circuit Boards with Standard Office Printers & Inks". In: *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST '15*. the 28th Annual ACM Symposium. Daegu, Kyungpook, Republic of Korea: ACM Press, 2015, pp. 243–251. ISBN: 978-1-4503-3779-3. DOI: 10.1145/2807442.2807511. URL: http://dl.acm.org/citation.cfm?doid=2807442.2807511.

[17] Joshua Chan, Tarun Pondicherry, and Paulo Blikstein. *LightUp: An augmented learning platform for electronics.*

[18] *Contact, Spring Loaded Pin, Radius, 2 A, 7.26 mm, 25 g.* URL: https://nl.farnell.com/mill-max/0906-3-15-20-75-14-11-0/spring-loaded-pin-2a-5-51mm/dp/2751178 (visited on 08/27/2019).

[19] Susan Cotterell et al. "eBlocks: An Enabling Technology for Basic Sensor Based Systems". In: *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*. IPSN '05. event-place: Los Angeles, California. Piscataway, NJ, USA: IEEE Press, 2005. ISBN: 978-0-7803-9202-1. URL: http://dl.acm.org/citation.cfm?id=1147685.1147754.

[20] *DS2480B Serial to 1-Wire Line Driver - Maxim.* URL: https://www.maximintegrated.com/en/products/interface/controllers-expanders/DS2480B.html (visited on 08/22/2019).

[21] Circuit Basics {\textbar} DIY Electronics {\textbar} 45. *Basics of the SPI Communication Protocol.* Circuit Basics. Feb. 13, 2016. URL: http://www.circuitbasics.com/basics-of-the-spi-communication-protocol/ (visited on 08/22/2019).

[22] *ESP8266 Overview | Espressif Systems.* URL: https://www.espressif.com/en/products/hardware/esp8266ex/overview (visited on 08/22/2019).

[23] Node js Foundation. *Node.js.* Node.js. URL: https://nodejs.org/en/ (visited on 08/22/2019).

[24] Natalie Freed et al. "Sticking together: handcrafting personalized communication interfaces". In: *Proceedings of the 10th International Conference on Interaction Design and Children - IDC '11*. the 10th International Conference. Ann Arbor, Michigan: ACM Press, 2011, pp. 238–241. ISBN: 978-1-4503-0751-2. DOI: 10.1145/1999030.1999071. URL: http://portal.acm.org/citation.cfm?doid=1999030.1999071.

[25] M.M. Gorlick. "Electric suspenders: a fabric power bus and data network for wearable digital devices". In: *Digest of Papers. Third International Symposium on Wearable Computers.* Digest of Papers. Third International Symposium on Wearable Computers. San Francisco, CA, USA: IEEE Comput. Soc, 1999, pp. 114–121. ISBN: 978-0-7695-0428-5. DOI: 10.1109/ISWC.1999.806684. URL: http://ieeexplore.ieee.org/document/806684/.

[26] Saul Greenberg and Chester Fitchett. "Phidgets: Easy Development of Physical Interfaces Through Physical Widgets". In: *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*. UIST '01. event-place: Orlando, Florida. New York, NY, USA: ACM, 2001, pp. 209–218. ISBN: 978-1-58113-438-4. DOI: 10.1145/502348.502388. URL: http://doi.acm.org/10.1145/502348.502388.

[27] Daniel Groeger and Jürgen Steimle. "LASEC: Instant Fabrication of Stretchable Circuits Using a Laser Cutter". In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*. the 2019 CHI Conference. Glasgow, Scotland Uk: ACM Press, 2019, pp. 1–14. ISBN: 978-1-4503-5970-2. DOI: 10.1145/3290605.3300929. URL: http://dl.acm.org/citation.cfm?doid=3290605.3300929.

[28] Daniel Groeger and Jürgen Steimle. "ObjectSkin: Augmenting Everyday Objects with Hydroprinted Touch Sensors and Displays". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.4 (Jan. 8, 2018), pp. 1–23. ISSN: 24749567. DOI: 10.1145/3161165. URL: http://dl.acm.org/citation.cfm?doid=3178157.3161165 (visited on 08/27/2019).

[29] Nur Al-huda Hamdan, Simon Voelker, and Jan Borchers. "Sketch&Stitch: Interactive Embroidery for E-textiles". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. the 2018 CHI Conference. Montreal QC, Canada: ACM Press, 2018, pp. 1–13. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3173656. URL: http://dl.acm.org/citation.cfm?doid=3173574.3173656.

[30] Björn Hartmann et al. "Reflective physical prototyping through integrated design, test, and analysis". In: *Proceedings of the 19th annual ACM symposium on User interface software and technology - UIST '06*. the 19th annual ACM symposium. Montreux, Switzerland: ACM Press, 2006, p. 299. ISBN: 978-1-59593-313-3. DOI: 10.1145/1166253.1166300. URL: http://dl.acm.org/citation.cfm?doid=1166253.1166300.

[31] Steve Hodges et al. "Circuit Stickers: Peel-and-stick Construction of Interactive Electronic Prototypes". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '14. event-place: Toronto, Ontario, Canada. New York, NY, USA: ACM, 2014, pp. 1743–1746. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557150. URL: http://doi.acm.org/10.1145/2556288.2557150.

[32] Steve Hodges et al. ".NET gadgeteer: a new platform for K-12 computer science education". In: *Proceeding of the 44th ACM technical symposium on Computer science education - SIGCSE '13*. Proceeding of the 44th ACM technical symposium. Denver, Colorado, USA: ACM Press, 2013, p. 391. ISBN: 978-1-4503-1868-6. DOI: 10.1145/2445196.2445315. URL: http://dl.acm.org/citation.cfm?doid=2445196.2445315.

[33] *I2C - What's That?* I2C Bus. URL: /http://www.i2c-bus.org/i2c-bus/ (visited on 08/22/2019).

[34] IFTTT. *IFTTT*. URL: https://ifttt.com (visited on 08/28/2019).

[35] Yoshihiro Kawahara et al. "Instant inkjet circuits: lab-based inkjet printing to support rapid prototyping of UbiComp devices". In: *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13*. the 2013 ACM international joint conference. Zurich, Switzerland:

ACM Press, 2013, p. 363. ISBN: 978-1-4503-1770-2. DOI: 10.1145/2493432.2493486. URL: http://dl.acm.org/citation.cfm?doid=2493432.2493486.

[36] Majeed Kazemitabaar et al. "MakerWear: A Tangible Approach to Interactive Wearable Creation for Children". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*. the 2017 CHI Conference. Denver, Colorado, USA: ACM Press, 2017, pp. 133–145. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025887. URL: http://dl.acm.org/citation.cfm?doid=3025453.3025887.

[37] Kristof Van Laerhoven et al. "Pin&Play: The Surface as Network Medium". In: *IEEE Communications Magazine* (2003), p. 6.

[38] Mannu Lambrichts, Jose Maria Tijerina Munoz, and Raf Ramakers. "SoftMod: A Soft Modular Plug-and-Play Kit for Prototyping Electronic Systems". In: *(Submitted to TEI 2020). Sydney, Australia*. 2020.

[39] Mannu Lambrichts et al. "DIY Fabrication of High Performance Multi-Layered Flexible PCBs." In: *(Submitted to UIST '19 Extended Abstract/Poster). New York, NY, USA*. 2019.

[40] Johnny C. Lee et al. "The calder toolkit: wired and wireless components for rapidly prototyping interactive devices". In: *Proceedings of the 2004 conference on Designing interactive systems processes, practices, methods, and techniques - DIS '04*. the 2004 conference. Cambridge, MA, USA: ACM Press, 2004, p. 167. ISBN: 978-1-58113-787-3. DOI: 10.1145/1013115.1013139. URL: http://portal.acm.org/citation.cfm?doid=1013115.1013139.

[41] David I. Lehn et al. "e-TAGs: e-Textile Attached Gadgets". In: 2004.

[42] *Light_ WS2812 library V2.0 – Part I: Understanding the WS2812 – Tim's Blog*. URL: https://cpldcpu.wordpress.com/2014/01/14/light_ws2812-library-v2-0-part-i-understanding-the-ws2812/ (visited on 08/22/2019).

[43] *Li-Polymer Battery 3.7V 190mAh*. MikroElektronika. URL: http://www.mikroe.com/li-polymer-battery-37v-190mah (visited on 08/22/2019).

[44] Joanne Lo and Eric Paulos. "ShrinkyCircuits: sketching, shrinking, and formgiving for electronic circuits". In: *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*. the 27th annual ACM symposium. Honolulu, Hawaii, USA: ACM Press, 2014, pp. 291–299. ISBN: 978-1-4503-3069-5. DOI: 10.1145/2642918.2647421. URL: http://dl.acm.org/citation.cfm?doid=2642918.2647421.

[45] *Makeblock: Global STEAM Education Solution Provider*. Makeblock. URL: https://www.makeblock.com/ (visited on 08/22/2019).

[46] *MCP9700T-E/LT Microchip Technology | Sensors, transducers | DigiKey*. URL: https://www.digikey.be/product-detail/nl/microchip-technology/MCP9700T-E-LT/MCP9700T-E-LTCT-ND/736010 (visited on 08/22/2019).

[47] David A. Mellis et al. "Microcontrollers as material: crafting circuits with paper, conductive ink, electronic components, and an "untoolkit"". In: *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction - TEI '13*. the 7th International Conference. Barcelona, Spain: ACM Press, 2013, p. 83. ISBN: 978-1-4503-1898-3. DOI: 10.1145/2460625.2460638. URL: http://dl.acm.org/citation.cfm?doid=2460625.2460638.

[48] Amon Millner and Edward Baafi. “Modkit: Blending and Extending Approachable Platforms for Creating Computer Programs and Interactive Objects”. In: *Proceedings of the 10th International Conference on Interaction Design and Children*. IDC ’11. event-place: Ann Arbor, Michigan. New York, NY, USA: ACM, 2011, pp. 250–253. ISBN: 978-1-4503-0751-2. DOI: 10.1145/1999030.1999074. URL: http://doi.acm.org/10.1145/1999030.1999074.

[49] *MINDSTORMS® | Themes | Official LEGO® Shop US*. URL: https://www.lego.com/en-us/themes/mindstorms (visited on 08/28/2019).

[50] Stefanie Mueller, Bastian Kruck, and Patrick Baudisch. “LaserOrigami: Laser-cutting 3D Objects”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’13. event-place: Paris, France. New York, NY, USA: ACM, 2013, pp. 2585–2592. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481358. URL: http://doi.acm.org/10.1145/2470654.2481358.

[51] Steven Nagels et al. “Silicone Devices: A Scalable DIY Approach for Fabricating Self-Contained Multi-Layered Soft Circuits using Microfluidics”. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI ’18*. the 2018 CHI Conference. Montreal QC, Canada: ACM Press, 2018, pp. 1–13. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3173762. URL: http://dl.acm.org/citation.cfm?doid=3173574.3173762.

[52] Grace Ngai et al. “I*CATch: A Scalable Plug-n-play Wearable Computing Framework for Novices and Children”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’10. event-place: Atlanta, Georgia, USA. New York, NY, USA: ACM, 2010, pp. 443–452. ISBN: 978-1-60558-929-9. DOI: 10.1145/1753326.1753393. URL: http://doi.acm.org/10.1145/1753326.1753393.

[53] Grace Ngai et al. “The TeeBoard: an education-friendly construction platform for e-textiles and wearable computing”. In: *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*. the SIGCHI Conference. Boston, MA, USA: ACM Press, 2009, p. 249. ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518742. URL: http://dl.acm.org/citation.cfm?doid=1518701.1518742.

[54] *Node-RED*. URL: https://nodered.org/ (visited on 08/22/2019).

[55] *Particle*. Particle. URL: https://www.particle.io/ (visited on 08/28/2019).

[56] Hannah Perner-Wilson, Leah Buechley, and Mika Satomi. “Handcrafting textile interfaces from a kit-of-no-parts”. In: *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction - TEI ’11*. the fifth international conference. Funchal, Portugal: ACM Press, 2011, p. 61. ISBN: 978-1-4503-0478-8. DOI: 10.1145/1935701.1935715. URL: http://portal.acm.org/citation.cfm?doid=1935701.1935715.

[57] Jie Qi and Leah Buechley. “Electronic Popables: Exploring Paper-based Computing Through an Interactive Pop-up Book”. In: *Proceedings of the Fourth International Conference on Tangible, Embedded, and Embodied Interaction*. TEI ’10. event-place: Cambridge, Massachusetts, USA. New York, NY, USA: ACM, 2010, pp. 121–128. ISBN: 978-1-60558-841-4. DOI: 10.1145/1709886.1709909. URL: http://doi.acm.org/10.1145/1709886.1709909.

[58] Jie Qi and Leah Buechley. "Sketching in Circuits: Designing and Building Electronics on Paper". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '14. event-place: Toronto, Ontario, Canada. New York, NY, USA: ACM, 2014, pp. 1713–1722. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557391. URL: http://doi.acm.org/10.1145/2556288.2557391.

[59] Raf Ramakers, Kashyap Todi, and Kris Luyten. "PaperPulse: An Integrated Approach for Embedding Electronics in Paper Designs". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. the 33rd Annual ACM Conference. Seoul, Republic of Korea: ACM Press, 2015, pp. 2457–2466. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702487. URL: http://dl.acm.org/citation.cfm?doid=2702123.2702487.

[60] *S4A*. URL: http://s4a.cat/ (visited on 08/22/2019).

[61] Valkyrie Savage, Xiaohan Zhang, and Björn Hartmann. "Midas: fabricating custom capacitive touch sensors to prototype interactive objects". In: *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. the 25th annual ACM symposium. Cambridge, Massachusetts, USA: ACM Press, 2012, p. 579. ISBN: 978-1-4503-1580-7. DOI: 10.1145/2380116.2380189. URL: http://dl.acm.org/citation.cfm?doid=2380116.2380189.

[62] Eric Schweikardt and Mark D. Gross. "roBlocks: a robotic construction kit for mathematics and science education". In: *Proceedings of the 8th international conference on Multimodal interfaces - ICMI '06*. the 8th international conference. Banff, Alberta, Canada: ACM Press, 2006, p. 72. ISBN: 978-1-59593-541-0. DOI: 10.1145/1180995.1181010. URL: http://portal.acm.org/citation.cfm?doid=1180995.1181010.

[63] *SONOFF Official Homepage - Smart Home Automation SONOFF Products*. SONOFF Official. URL: https://sonoff.tech/ (visited on 08/28/2019).

[64] Becky Stern and Tayler Cooper. *Getting started with Adafruit FLORA: making wearables with an Arduino-compatible electronics platform*. Maker Media, Inc, 2015.

[65] *TC1262 - Linear Regulators*. URL: https://www.microchip.com/wwwproducts/en/TC1262 (visited on 08/22/2019).

[66] *Teach, Learn, and Make with Raspberry Pi – Raspberry Pi*. URL: https://www.raspberrypi.org (visited on 08/22/2019).

[67] *The Official Arduino AVR core*. original-date: 2017-09-11T14:43:13Z. Aug. 20, 2019. URL: https://github.com/arduino/ArduinoCore-avr (visited on 08/22/2019).

[68] Udayan Umapathi et al. "LaserStacker: Fabricating 3D Objects by Laser Cutting and Welding". In: *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. UIST '15. event-place: Charlotte, NC, USA. New York, NY, USA: ACM, 2015, pp. 575–582. ISBN: 978-1-4503-3779-3. DOI: 10.1145/2807442.2807512. URL: http://doi.acm.org/10.1145/2807442.2807512.

[69] Tom Valkeneers, Danny Leen, and Raf Ramakers. "Stack-Mold: Rapid Prototyping of Functional Multi-Material Objects with SelectiveLevels of Surface Details." In: *InProceedings of the 32nd Annual ACM Symposium onUser Interface Software and Technology (UIST '19). New York, NY, USA*. 32nd Annual ACM Symposium. 2019.

[70] Nicolas Villar et al. “The VoodooIO gaming kit: a real-time adaptable gaming controller”. In: *Computers in Entertainment* 5.3 (July 1, 2007), p. 7. ISSN: 15443574. DOI: 10.1145/1316511.1316518. URL: http://portal.acm.org/citation.cfm?doid=1316511.1316518 (visited on 08/22/2019).

[71] *Vishay - Capacitors - VJ....W1BC Basic Commodity Series - Surface Mount Multilayer Ceramic Chip Capacitors for Commodity Applications*. URL: http://www.vishay.com/capacitors/list/product-28548/ (visited on 08/22/2019).

[72] Martin Weigel et al. “iSkin: Flexible, Stretchable and Visually Customizable On-Body Touch Sensors for Mobile Computing”. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. the 33rd Annual ACM Conference. Seoul, Republic of Korea: ACM Press, 2015, pp. 2991–3000. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702391. URL: http://dl.acm.org/citation.cfm?doid=2702123.2702391.

[73] Michael Wessely, Theophanis Tsandilas, and Wendy E. Mackay. “Stretchis: Fabricating Highly Stretchable User Interfaces”. In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16*. the 29th Annual Symposium. Tokyo, Japan: ACM Press, 2016, pp. 697–704. ISBN: 978-1-4503-4189-9. DOI: 10.1145/2984511.2984521. URL: http://dl.acm.org/citation.cfm?doid=2984511.2984521.

[74] *Wi-Fi Standalone | Microchip Technology*. URL: https://www.microchip.com/design-centers/wireless-connectivity/embedded-wi-fi/wi-fi-standalone (visited on 08/28/2019).

[75] *WS2812C-2020 WORLDSEMI - LED | TME - Electronic components*. URL: https://www.tme.eu/en/details/ws2812c-2020/smd-colour-leds/worldsemi/ (visited on 08/22/2019).

[76] Ruquan Ye, Dustin K. James, and James M. Tour. “Laser-Induced Graphene: From Discovery to Translation”. In: *Advanced Materials* 31.1 (Jan. 2019), p. 1803621. ISSN: 09359648. DOI: 10.1002/adma.201803621. URL: http://doi.wiley.com/10.1002/adma.201803621 (visited on 08/27/2019).

[77] Jaeyoung Yoon et al. “Soft Modular Electronic Blocks (SMEBs): A Strategy for Tailored Wearable Health-Monitoring Systems”. In: *Advanced Science* 6.5 (Mar. 1, 2019), p. 1801682. ISSN: 2198-3844. DOI: 10.1002/advs.201801682. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/advs.201801682 (visited on 08/28/2019).