

DAVID CUARTIELLES PLATFORM DESIGN

Creating Meaningful Toolboxes When People Meet

TEKHNOLOGIA OMNIPOTENS REGNAT

•XIII•



MALMÖ
UNIVERSITY

PLATFORM DESIGN

Dissertation Series: New Media, Public Spheres and Forms of Expression

Doctoral dissertation in Interaction Design

Faculty: Culture and Society

Department: School of Arts and Communication, K3

Malmö University

Information about time and place of public defense, and electronic
version of the dissertation: <http://hdl.handle.net/2043/26130>

CC BY-NC-SA David Cuartielles Ruiz, 2018

Copy editor (chapters 1-6): Charlotte Merton

Illustration*: Julia García

Cover, bleed design and layout: Laura Balboa

*Adapted from original artwork by Boamistura

ISBN 978-91-7104-942-1 (print)

ISSN 978-91-7104-943-8 (pdf)

Printed by Holmbergs, Malmö 2018

This work is licensed under the Creative Commons Attribution-
NonCommercial-ShareAlike 4.0 International License. To view a copy
of this license, visit <http://creativecommons.org/licenses/by-nc-sa/4.0/>
or send a letter to Creative Commons, PO Box 1866, Mountain View,
CA 94042, USA.

DAVID CUARTIELLES

PLATFORM DESIGN

Creating Meaningful Toolboxes When People Meet

Malmö University, 2018

Para Bobbie. Ahora vámonos de fiesta.

ABSTRACT

Platform Design is a study of different viewpoints on the creation of digital systems, and how they converge in platforms designed, built, and managed by communities. As sociotechnical constructs in which features emerge through the interaction of different stakeholders, platforms are understood as both means and outcomes—the ‘things’ or boundary objects in a design process—generating the spaces where communities of practice can form.

Utilizing two strongly interwoven timelines in education and research (both in academia and industry), the thesis shifts the centre of balance in actor–networks by iteratively recalibrating from a techno-deterministic analysis towards a community-driven one. The theoretical background in the fields of cybernetics, critical theory, design, and the sociology of technology frames the empirical work, which consists of academic publications, design reports, and the publicly available documentation of realized projects. In the space between theory and praxis, a methodological toolbox is developed, *a posteriori* revisiting experiences gathered over a decade Drawing on a series of functional concepts, the thesis proposes an alternative co-design framework, termed *inclusive multiple prototyping*. Meant to augment new sensibilities that are pertinent to the design process of platforms, this framework addresses the inherent complexity of actor–networks and human–machine communities.

In practical terms, the thesis describes a series of projects, some of which can be considered platforms, while others would be better categorized as tools, toolboxes, kits, or infrastructure. These include co-creating the Arduino community, repurposing kitchen appliances for connection to the cloud, designing a modular prototyping platform involving programming and electronics, deploying an indoor location system, creating educational kits for upper secondary school teachers, and inventing new haptic interactive interfaces. Some of the

projects required the long-term involvement of the researcher in intimate communities of practice; others were temporal interventions, yet reached thousands of users. Practice-based and transdisciplinary, the thesis contributes to the field of interaction design by bringing in elements of a sociotechnical discourse, while problematizing notions such as democracy and governance, openness of tools and outcomes, modularity, generalizability, and transferability—the three latter terms further fuelling the research questions. The research shows that these are properties that enable the creation of platforms, although the question remains whether there is such a thing as a standardized platform. While this thesis touches upon the potentials of state-of-the-art platform technology, it also points to the fact that there is work to be done, socially, ethically, and politically, when considering the augmentation of platforms for everyday use as pervasive and artificial intelligence agents.

ACKNOWLEDGEMENTS

This has been a ride. Ricardo, from the Colaborabora collective in Bilbao told me once that everything about my work is epic ... that I make it epic through my way of telling the story, but to be fair, things have truly been epic for the last 15 years of my life ... from the day when I started evaluating Anoto™ pens with Lone Malmborg at the ‘Micromobility and Learning’ project, to the night prior to sending this text to the printer by the hands of Laura Balboa, every minute has been packed with impossible deadlines, long development nights, experiments with hundreds (and thousands) of users, hacks on remote servers to simulate connected kitchens, dawns with orange skies at the university lab, coding at airports all over the world, and passing out to intricate French sociologists on every sofa in Sweden.

The projects in this thesis have brought me to places I had never imaged before and opened the door to even more projects, making even more designs and meeting even more people. When it comes to people, I have collaborated, assisted, and being helped by so many individuals, collectives, and companies that there is not enough ink in my printer to write down some kind words about every one of you for the time you invested in helping me making this thesis.

First and foremost I want to thank Maria Hellström Reimer and Daniel Spikol, for challenging my thought during the last two years and helping me clean up the noise away from the signal. I don’t know how many times I heard the sentence “this is interesting, but it can wait until you make your next thesis” referring to such a broad range of different things; from short stories about the interaction in the Arduino community to the whole life story of electronic musicians were at some point parts of this thesis and it is only thank to Maria and Daniel that this whole thing ended up making sense.

Other professors and heads of research groups have also been there at different parts of the process, especially Lone Malmborg who opened the door to this world for me and Jonas Löwgren who managed to keep a poker face through the longest list of weird thesis titles over many years. Also from Malmö University, I want to thank Bo Peterson for hosting the Arduino meetings at Malmö University between 2007 and 2009 when my daughter was too small for me to travel around. I will never forget the meetings at the old K3 building during the Swedish höstlov when so many things of our little open source project were decided. In that same line, Simon Niedenthal during his time as program lead in interaction design and school administrator Paula Pragert contributed to making my first lab from an administrative and a financial point of view. Without their support, I would have never had the chance to experiment with electronics in the classroom. I keep in a folder the first drawings I made to explain the school's board about which were the parts needed to build an electronics lab, something of a challenge when being at a school that had created the Bauhaus manifesto for the XXI century, where everything was going to be digital, pervasive, and ubiquitous, but also immaterial. And that brings me to thank Pelle Ehn, probably the most recognizable author of that manifesto that brought me to Sweden. Pelle has shown me, through his example, that curiosity has no age, and that designers need to keep a smile on their face under any circumstances if they want to be able of facilitating design.

My work has been partially financed by different research groups: Creative Environments, Medea, and lately IOTAP. Within this context I would like to mention Paul Davidsson, head of IOTAP, who made some room for me the last years and that opened the possibility of working with design and technology at once. But also Sara Bjärstorp, at the time of writing Head of K3, in representation of herself but also of the previous directors of the institution that offered me the possibility to continue with the thesis work on the side to my position as lecturer.

There are many friends and colleagues at the School of Arts and Communication (K3) that I would like to remember through these lines. Mattias Nordberg, for whom crafts have no secret, the man

who makes a chair a year, and that made so many of my projects possible. Kristina Regnell who managed to remember the names of all of the students at the school from the time when we had ten students per class to the year we had 65.

The Prototyping Laboratory I created in 2003 has now changed name to the Institute of Interactive Objects but keeps the same spirit behind it of having students getting in control of the lab and making it a space for collaboration and integration of the different education programmes at school. There are plenty of alumni that took the responsibility of running the lab, being Marcus Hannerstig and Pontus Stalin the two representatives of the first generation of students. Together with them was Tomek, our eternal German student and creator of the first DMX library for Arduino, and Marcos Yarza who came all the way from Zaragoza to help us making some of the first engineering experiments with Arduino and who became head engineer of a company that started selling wireless augmentation boards for Arduino. Tony Olsson and Andreas Göransson came in a couple of years later and stayed at the university for a long period transitioning from students to teachers. Tony, thanks for being there all of these years, from taking over some of my classes to flying to Mexico to run a lecture with professors at the Tecnológico de Monterrey on my name, to being the camera man for some of my early Arduino projects, you did so much that I doubt I would be here without you; Andreas thanks for being so patient when we made the two least successful Android programming books together, but more importantly, I will always be grateful for helping out transforming the programming classes at K3 into something new and exciting. I would like to thank Anuradha Reddy, my late PhD conspirator, for always looking at whatever strange text came from me with fresh eyes, and for the long conversations about design vs. engineering.

Most of my projects outside the university have been possible thanks to the cooperation of public institutions, foundations, and the people behind them. Marcos and Laura at *Medialab Prado* (formerly known as Medialab Madrid) opened the possibility of meeting other institutions and participating in Ars Electronica 2006. Also at that time, I got to know Eva Gómez who invited me to run the first Arduino

workshop in México in 2007, also thanks to her and Jesús Oyamburu (back then director of the *Centro Cultural de España en México*) I got access to a grant that allowed me work at FARO de Oriente in México City for several years. Even if I didn't have the chance to mention much about that work in the thesis, making that project taught me a lot about how to deal with complex social contexts, and for that I will always be grateful.

While I did run some experiments in education before 2012, it was then when Jesús Fernández Cid at the *Centro de Recursos del Professorado de Castilla La Mancha* offered me the possibility of making a pilot with 25 schools, almost 700 students, what really marked a difference in the way I understood how to deal with large size prototypes. That project was taken over by the *La Caixa foundation*, a non-profit where Javier Hidalgo, Antonio García, and Ester Arderiu helped me and my team reach tens of thousands of students in four different regions. It should also be mentioned the help offered by Alexander Flor and his team at the *Telefonica foundation Ecuador* who helped bringing one of my educational experiments to 40 of the schools supported by them in the country. At the city of Zaragoza, Spain, I had the pleasure of working with José Carlos Arnal and the *Fundación Zaragoza Ciudad del Conocimiento* in designing a technology summer camp for kids aged 7 to 14; I should extend my gratitude to the whole team that made that project possible, specially Carmen Marín, who acted as project manager and made sure everything was running as planned, and David Gascueña 'Gasku' who hacked the most expensive water fountain in history (sure this is an overstatement, but it was impressive) using open source technology.

I did not only have institutional collaborations over the years, I also worked hand by hand with engineers, designers, and clerks from different companies in enhancing some of my projects, but mainly in convincing them that open source technology and free software are a great contribution to human kind. Among the most relevant collaborations I should mention the one with Telefonica's R&D team lead by Fco. Javier Zorzano, who put a skilled team of people in making some of the GSM boards for Arduino, and designed an early version of the protocol that would run the modular electronic boards used in

the PELARS project. At a different branch of Telefonica (*Telefónica Educación Digital*), Yuma Inzolia Berardi and her team gave me the possibility of creating a national contest in technology and education that happened in Spain in 2016 and 2017.

But going back to academia, I have to thank a long list of people that came to the rescue with last minute references and books that “should for sure be there” and that kept me awake for days and weeks while trying to figure out how to get those into the final manuscript. Paz Sastre introduced me to Bratton over a coffee in México City. José Saldaña, old colleague from my times as an engineering student, briefly talked to me about his work one day we met by accident at a corridor at the university and one of his papers helped me articulating a large portion of Chapter 2. Stahl Stenslie who made all of the projects I would like to do but I never dared, showed up one day at my lab and offered collaborating in some of the papers that have become part of the compilation. Moisés Mañas and María José Martínez de Pisón always supported me from their *Laboratorio de Luz* and gave me some of the basic references in the field of arts that I needed to bring in creativity to most of my projects. The day I presented my 90% seminar, Prof. Bo Reimer gave me three references on platform studies that became key for the final thesis, I guess I have learned the lesson now, next time I should visit Bo first, he knows what he is talking about.

Roberto Casas, Álvaro Marco, Hector Gracia and Prof. Jorge Falcó deserve a special acknowledgement from the bottom of my heart. Their work in indoor positioning systems was ground-breaking at the time it was done and if opened the possibility for me to get into a technical project with my designer eyes. It was the first project I made where I had the opportunity to think about users and not bits. But nothing of that would have been possible if Roberto, Álvaro, Hector, and Jorge hadn’t had such a tremendous understanding of the technology we were dealing with.

When I started my research studies little I knew that I would end up co-producing one of the most successful electronic prototyping tools of our time, Arduino. That was possible only thanks to those that

were there making it possible on a day to day basis: Massimo Banzi, Dave Mellis, Tom Igoe, and Gianluca Martino. Thinks might not have always been colourful among us, but we made it, and nobody else. We put our knowledge, our time, and our resources into making something that is bigger than us, and that allowed us getting to know more people, help science advance, and make our digital world a little more open. At Arduino I want to thank the teams in Torino and Malmö for giving me space to finish this thesis, a work that is as yours as mine: Nerea, Fabio, Ernesto, Sara, Kristoffer, Carlos, Jake, Pietro, Daria, Martino, Mr. Vacuum, Cristina, Zoltan, Csaba, Fabrizio ... you are so many I cannot even remember all of the names. I am grateful to the team that stayed over the years, but also to those that left to pursue their own plans. I am especially thankful to those that helped making CTC, Arduino's first education programme, as well as the Arduino robot: Xun Yang, and Tien Pham. There is also a special place in my heart for Davey Taylor, a true hacker making no compromises. Among the ones I will never forget is Quinn Ertel, who simplified my life by an order of magnitude. Beyond those that are part of the Arduino company, there is also 26 million Arduino users I am grateful, too. Please keep on doing what you are doing: build, program, compile, burn, iterate.

I should send a big hug my extended family at the *Ashoka* foundation and the one I am generating thanks to their support: Jose García, the guys at *Instroniks*, Marta and Marga with their robotics for kids, and Julia García who came out of nowhere to make the best cover illustration ever for this thesis. Also, to the people at the *Puig foundation* that are enabling for this new project to happen.

Finally I want to thank my family: Malva and Joaquín for insisting year after year that I should get this done, to Diego who is there even when far away and that reminded me about Boamistura's mural painting that ended up illustrating the book, to Bobbie for all that time that I stole from you to make this happen, and to Laura my partner in life and projects, the one that could crack jokes while correcting my grammar and making the layout of this book, without you I wouldn't have made it ... thank you. And now, let's take some time off.

CONTENTS

ABSTRACT	5
ACKNOWLEDGEMENTS	7
1 INTRODUCTION	19
Background	21
Research questions	36
Limitations	37
Summary.....	39
2 THEORETICAL FRAMEWORK	45
From networks to platforms	45
Preconceptions	60
Summary.....	83
3 PAPERS	87
The compilation in context	87
Resign desearch	91
Hidden issues in deploying an indoor location system	95
EU reports as design diaries	96
Haptics	103
Summary.....	108
4 METHODOLOGY	113
Reflection in action	114
Openness	119
Use, participation, action, activism	124
Inclusive multiple prototyping	132
Concluding reflections on mixed methods	134

5 LESSONS LEARNT	139
Papers and other projects	141
Overall concepts	149
Summary	170
6 CONCLUSIONS AND REFLECTIONS.....	175
The platform-thing	178
Visions of possible futures	180
REFERENCES	191

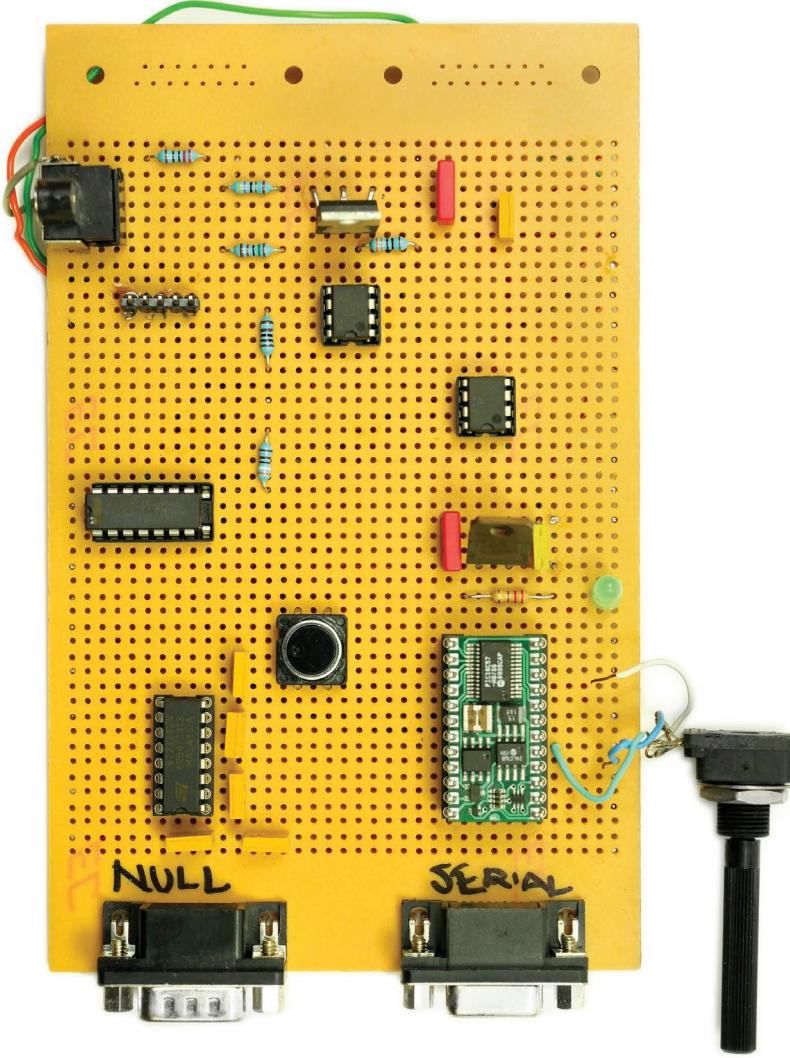


FIGURE 1
HANDMADE PROTOTYPING BOARD, 2003

1 INTRODUCTION

At the turn of our century, the miniaturization of technology allowed digital artefacts to be embedded into almost anything. Centimetre-sized computers can capture and transmit data to other computation units for further processing, but also affect the physical world by providing feedback, however cursory. This miniaturization of sensors, actuators, and communicators simultaneously realized the vision of pervasive and ubiquitous computing (Weiser 1991; Mattern 2005), two paradigms that had conceptualized technology as both embedded in the environment and embodied in humans. With the arrival of other technologies, such as radio-frequency identification or RFID (Hayles 2009) or crypto-chips (Hafner et al. 1991), everyday objects could be assigned unique identification numbers in order to digitize, monitor, and process their unique interactions.¹ The convergence of identification, sensing/actuating, and communicating technology into things was the computing paradigm of the Internet of things (IoT) (Greengard 2015). This new understanding of the (digital) world established a situation where always-on, always-connected objects provide endless streams of information (often referred to as big data) to computational machines that will search for patterns and events (Brody & Pureswaran 2014). Information flows happen at such a pace that it will never be possible for humans to compete with machines in processing data and, given new techniques like machine learning (ML) or deep learning (DL), not even in data analysis (Rouvroy 2013; Boden 2017).

This raises questions about the role of humans in these new systems—systems where machines are responsible for determining the information to be extracted and the events that happen in data flows. Also, if everything can be digitized, even human interaction, what is the mechanism for those interactions to happen? Digital platforms, as

¹ RFID stands for radio-frequency identification

will be explained later, are featureless systems where humans and non-humans engage in relationships (Gillespie 2010; Helmond 2015). The exchange of information between parties is effected through pervasive devices that transfer data—sometimes with human consent, sometimes not—to the huge spaces of computation, data processing, and storage that we call platforms, a definition that will be challenged in this thesis. More and more of our new reality is built on (and through) platforms of various types (Helmond 2015). Therefore, such systems, their creation, and the involvement of human and machine participants in the process have become a relevant field of study (Plantin et al. 2016). This thesis focuses on the human relationships in the co-creation of platforms, but also on identifying a series of characteristics that could be assigned to platforms as they evolve from concept to tool, and from there to boundary object in a network of relationships.

Platform Design is a compilation thesis that addresses the theory, practice, and method seen in the co-creation of various forms of digital platform over the course of fifteen years. During this time I participated in multiple projects, of which four are part of the compilation, while others are introduced in the summary chapters to highlight the value of the side products of the research. The introduction and the analysis of the papers in the compilation are constructed chronologically, because of the importance that the elapse of time played in the development of the research. All the projects presented here resulted in prototypes—some large, reaching thousands of users—that after testing gave me valuable leads for subsequent experiments and projects, and also contributed to the visions and theories registered in this thesis.

The text is divided in six chapters, starting with the introduction, which presents the background, research question, and limitations to the overall process. Chapter 2 describes the theoretical framework of the thesis, starting with a series of observations of terms that evolved over time, offering an understanding of the field as the balance of various pairs of values, arranged as axes in a multidimensional space of meaning. This is followed by a series of definitions of the most important preconceptions, which inform my discussion of what any

designer should bear in mind when designing a platform. Chapter 3 sets out the papers included in the compilation in chronological order. Chapter 4 charts the methodological framework, which was constructed parallel with the development of the thesis—a series of mixed methods, including traditional metrics, social and legal contracts, activist research, and the creation of *inclusive multiple prototypes*. Chapter 5 reviews all the key projects conducted over the years (not just the ones in the compilation) through the chosen methodological lens, resulting in a conceptual framework fashioned using new terminology that reorients the thinking in platform design away from the techno-deterministic and towards the human-centric; this framework is one of the significant contributions made by this thesis. Having initially observed the properties of size, performance, and pace in my research, as projects unfolded I discovered a very different set of values to take into account when creating platforms with a co-design process. Chapter 6 reviews the research question, draws a series of conclusions, and outlines future lines of research in platform design.

The papers that make up this compilation include a design manifesto; three research project reports, which could be considered design diaries of sorts; a technical journal article; and a series of short papers exploring the new field of haptics in user interface design. While there were other papers to consider when writing this thesis, the ones chosen for presentation offer a comprehensive overview of the work in the field. In the process of co-creating different platforms I played several roles, which generated different types of results: reports, diaries, annotated portfolios, academic articles, open-source software, blueprints for electronic printed circuit board designs, and so on. By intentionally selecting a range of outcomes to include in the compilation, this thesis underlines the mixed nature of the end results in contemporary academia.

Background

A study of platform creation, this thesis is based on my work as a practising educator and on my research as a member of various research groups at Malmö University and at the Swedish company

Arduino Verkstad AB. The work intermeshed over the years in a series of projects that involved academic colleagues, communities of users, and developers of different platforms. Separating out the teaching and research timelines has been a complex exercise, but offers readers a better understanding of the past processes involved; however, it was not the way I experienced things, the future being a closed book. This section contextualizes the course of my research, looking at my research questions, the actual research, the outcome in the form of the papers in this compilation, and how the cutting-edge technology evolved over time.

When I was first introduced to the field of interaction design (IxD) in 2000, it was as an assistant professor in the field of programming for designers. I planted the seed of what designers learn today in terms of software at Malmö University, disregarding many of the intricate aspects of programming in favour of new paradigms and metaphors to explain complex concepts to design and art students. Back then, most of the educational tools for software learning had been created by and for engineers, and teachers around the world struggled to create a curriculum that would accommodate the students' backgrounds while introducing them to the world of design with tools that were closer to the metal than traditional computer-aided design and graphic design tools.² It was our goal to teach interactivity through tools that would allow our students to become familiar with state-of-the-art technologies, but not push them into becoming programmers (even if some did indeed so).

One of the pioneering designers in the field was John Maeda, who with his publication *Design By Numbers* (1999) and eponymous software (DBN) presented a then revolutionary way of thinking about aesthetics and computation.³ A number of commercial tools for creators predated Maeda's—Mac OS software Hypercard being

² The 'metal' in digital electronics refers to microchips. There is a hierarchy of tools, with machine code the closest to the metal (since everything you write is exactly as it will be stored in the processor's memory). C, C++ and other programming languages are the next step out, and Java the next step again, since it is executed as a stream of code running on a virtual machine. Tools such as Photoshop or Audacity that handle media and have been written in high-level programming languages like C or Java are even further up in the abstraction ladder and therefore further away from the metal.

³ For the DBN timeline, see the official website at MIT Media Lab, <http://dbn.media.mit.edu/>.

one example—but I had never been exposed to any of those, and indeed I was unaware of Maeda’s work.⁴ It would not be until 2005 that he would enter in the picture, after I encountered the programming environment called Processing. Fry and Reas, the creators of Processing and former students of Maeda’s, explained in their 2007 book *Processing: A Programming Handbook for Visual Designers and Artists* how they created Processing ‘to teach fundamentals of computer programming within a visual context, to serve as a software sketchbook, and to be used as a production tool’. (Fry & Reas 2007, 1) Processing’s original user interface had clear references to Maeda’s DBN, having the same colour scheme, buttons, and visual distribution (Shiffman 2009). However, it also introduced a series of new metaphors, such as calling all of the programs ‘sketches’ and the folder containing them a ‘sketchbook’. Processing was first used with students back in 2001 (Fry & Reas 2007, xxv).

Parallel to my work teaching software, I gradually introduced the use of electronics to my students as a way to bring the IxD field closer to how devices are actually designed. I set up the so-called Prototyping Laboratory where I could introduce IxD students to sensor technology and the use of off-the-shelf microcontroller boards to create interactive systems. During 2003 and 2004, I experimented with commercial boards, and later with the creation of my own boards, to give student projects an afterlife beyond the original courses. Commercial boards then came as demonstrator boards, and featured a whole series of parts—buttons, screens, even small speakers—for students and professionals (and the end users of the boards) to experiment with the technology. These implementations rendered the actual functionality of the microchip effectively useless by removing access to many of the microcontrollers’ input/output pins, since they were being used with parts that were predetermined by the manufacturer of the boards. As a side effect, adding extra parts to the electronic designs made the boards unnecessarily expensive, which to all intents and purposes stopped the academic community from adopting any of the designs.

⁴ There are plenty of online resources about Hypercard, which turned out to be successful software. For example, <http://hypercard.org/> has multiple resources and links to related projects.

In the winter of 2003, and the early days of the experiments with my students at the Prototyping Laboratory, it dawned on me that there were aspects that many of the student projects had in common. There were also a series of features, or ‘affordances’ (Gibson 1986, 18; Norman 1990, 9), which the electronic boards had to have in order for students to be able to tackle as many design scenarios as possible.⁵ With that in mind, I ended up designing Malmö University’s first contribution to the world of tools for IxD: a DIY board that students had to assemble themselves before starting work on physical computing projects (see Fig. 1). It was a nameless board that students had to solder by hand to a standard electronic circuit. They had to prototype their projects on the boards, later making a second iteration where the students and I would manufacture a specific circuit board for each project. The intention behind this two-steps process was for students to learn the difference between proof of concept and prototype at the level of electronics. Whatever they did on the first handmade board was a proof of concept; the following steps were prototypes.

There was a democratic intentionality in making that first hand-mounted electronic board, as it meant that people who did not have an education in electronics were not excluded from the creative use of technology. The term ‘use’ should include the direct manipulation of digital things. Resnick et al. (1998) define digital manipulatives (DMs) as tools ‘with computational power embedded inside’ which ‘are designed to expand the range of concepts that children can explore through direct manipulation, enabling children to learn concepts that were previously considered “too advanced” for children’. Remove the word ‘children’ from that statement and it becomes a definition of what I was trying to do with those early prototyping boards. Through the ‘direct manipulation of the digital materials’, in design theorist Johan Redström’s terms (2001), IxD students could learn concepts that were previously considered too advanced for them.

⁵ Norman’s explanation is straightforward: ‘The term affordance refers to the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used ... Affordances provide strong clues to the operations of things ... When affordances are taken advantage of, the user knows what to do just by looking’ (1990, 9); Gibson, on the other hand, introduces the idea of affordances as ‘properties taken with reference to the observer’ (1986, 18).

I was not alone in this. In Japan in 2004, Kobayashi and colleagues adopted a similar process that ended with the creation of the Gainer platform as a way to introduce students to the creation of physical interfaces (Kobayashi et al. 2006). Another example of this type of educational process resulted with the creation of the Wiring platform at the Interaction Design Institute Ivrea (IDII) in Italy (Barragan 2004; Löwgren & Reimer 2013, 111). In other words, the idea of creating electronic platforms for prototyping emerged in different parts of the world at much the same time. All took a similar approach, minimizing the economic impact on the users while maximizing the possible uses of the boards and making the programming of the digital intelligence of the boards as simple as possible.

It was not a coincidence, then, that the Arduino platform was launched at this point. My interest in building digital tools for my students led me to join a number of international collaborations, one of which took me to the IDII in 2005. There I met Massimo Banzi and Dave Mellis, and later Tom Igoe and Gianluca Martino, with whom I co-founded the Arduino platform in the spring of 2005. Arduino was then nothing but an open-source tool for IxD and industrial design students to add electronic brains and interactive behaviours to their artefacts. It was an open-source hardware microcontroller board with an open-source programming environment that people could use to create their own prototypes and products using digital components. In March 2005, having discussed the ways in which our students and collaborators used electronics in their projects, Banzi and I designed the first Arduino board. Dave Mellis made the necessary software so that people could program their own interactive programs to run on the board. Dave and I also created the bootloader, the program residing in the microcontroller's memory that allows programs to upload to it from a computer. Our experience and the existing open-source software meant this could all be produced at speed. We were confident the students at IDII and Malmö University would find it useful, and we commissioned a local manufacturer to produce the first printed circuit boards, to be called Arduino (see Fig. 1).

Once the first boards were made, we introduced the tool as part of our standard curriculum at our respective universities. We developed

a toolbox of the interactive prototypes that included Arduino boards, electronic components, and open-source or freemium software that our students could use to create interactive devices. It was our understanding that in the not-so-distant future, digital artefacts would use screens to display information, so we built a series of examples using Flash—the now defunct software deployment tool from Macromedia and later Adobe—to show how to take advantage of computer screens to prototype interactions. At this point, Kobayashi (2006), Fry and Reas (2007) and others were publishing on the ways to connect microcontroller boards to Flash and Processing.⁶

The connection between software and hardware was a major part in Arduino's success as a platform. Microcontroller boards, like the ones by Arduino, are nothing but generic peripherals to computers, and can be used to do almost anything, from collecting information from the environment through sensors to affecting the same environment using motors. The Arduino project started with the creation of a microcontroller board that could be used to extend the capabilities of a personal computer towards the physical world. By 2005 personal computers had lost the ability to interface the physical world in order to read information from sensors or write to other actuators than the standard peripherals—keyboard, mouse, and screen. Arduino is a hardware tool designed to reverse this situation and reopen the connection between the physical and the digital worlds. Different communities of creators in the digital sphere use different software and hardware tools. For example, graphic designers typically use the Adobe Creative Suite, a package of digital tools, to process images both in raster and vector formats. Musicians use tools such as Ableton, a digital dashboard capable of communicating with other programs and physical interfaces using standard communication protocols. When creating the first Arduino hardware tools, we realized that one way to enhance the experience of learning about digital technology was to get the device—Arduino boards in this case—to send and receive data to existing software. By enabling the microcontroller hardware to communicate with software previously being

⁶ Fry and Reas' work connecting microcontrollers to computers was done earlier than ours at Arduino, but their first publication came out two years after we released the first Arduino board and some of the examples.

used by people and thus enhancing that software so it interfaced with the physical world, the learning curve of the Arduino boards was reduced, which resulted in a natural increase in their adoption rate.⁷

Arduino as a tool spread quickly. We were invited to lecture to a range of educational institutions, and this, along with my work with design and art students, brought home to me not just what was needed in terms of software and hardware, but also the types of documentation required in order to learn basic concepts and the standard parts people needed if they were to conduct basic experiments. This led me to the next conceptual shift. I understood that having a prototyping board—a tool—and the software that went with it—another tool—was not enough to learn about electronics. People would need a series of templates for meaningful activities and the electronic components necessary to replicate them and learn the basic aspects of the technology. Consequently, content is also very important in the process of democratizing knowledge: what I had lacked in the microcontroller platforms that preceded Arduino was content that would engage people in the topic. Maeda's DBN project included this notion of content in a form he called courseware, as defined on the project's website (Aesthetics and Computation Group 2003). Courseware introduces the idea of mixing course and software, and introduces new concepts in the form of progressive, self-explanatory exercises. Other programming languages such as Pure Data are by design self-explanatory due to their visual nature. However, unlike DBN's accompanying courseware or Pure Data's visual code, the world of microelectronics was not characterized by enticing, self-explanatory artefacts. On the contrary, it was presented in the form of datasheets and manuals.⁸ Datasheets (the way engineers share information about electronic components), and manuals (the way they explain software) were not appropriate for the average Arduino user back in 2005. We needed to develop examples that would interweave the building process with a more theoretical explanation.

7 In Arduino's early days in 2005, the main creative packages people were using were Pure Data and Max/MSP for sound and live interactive media production, Isadora for live video mixing, Processing for installations, and Flash for interactive web experiences.

8 Datasheets are the way information about electronic components and tools is presented to developers. There is no standard for how to present the information, because the nature of the components and tools varies from one to the next. Datasheets present the electrical qualities, electronic functionality, and mechanical properties of a component.

When we come new to a field, we do not typically possess the tools to operate in it; starting from scratch, we need to assemble a whole series of objects, concepts, contacts, and knowledge in what will become our own personal toolbox. This is what set me thinking about *kits* as conglomerates of tools, materials, and documentation. I experimented with kits, content, and produced outcomes in the shape of prototypes, courseware, and products in a range of fields: basic electronics (Gran et al. 2005; Cuartielles 2010, 2012), robotics (Cuartielles & Yang 2010; Cuartielles et al. 2011), and the integration of mobile devices and electronic prototypes (Göransson & Cuartielles 2012; Cuartielles & Göransson 2015), among others. While I was focused on putting kits in people's hands, I noticed that they were building their own collections of tools to suit their most common needs. For example, IxD students collected sensors to measure distance, accelerometers to detect movement, Peltier cells to heat or cool prototypes of wearable devices, and communicators—mainly Bluetooth wireless chipsets—to get their prototypes to communicate with mobile phones. Teachers in robotics at upper secondary schools, on the other hand, preferred light sensors, inexpensive DC motors, 10mm LEDs, and anything that might be handy when creating robots with their pupils. Each new group of Arduino users I encountered had different needs, all broadly similar to the others, but also different enough for me to realize that those collections of parts were characteristic of their skill sets and needs. It was these collections of tools and small parts that I identified as toolboxes. While kits are related to a context of use, toolboxes are related to a context of expertise. Expertise is acquired over time and is hard to come by in a new context with an already assembled toolbox. While kits are not the same thing as toolboxes, they can help us gain an understanding of what is needed and are thus indications of a toolbox's contents. The Arduino tools are open-ended enough to be included in both kits and toolboxes.

In time, Arduino grew from being a multiplicity of simple open-source toolboxes in the hands of various user groups, used for designing electronic products and experiences, to becoming a place where people came not only to learn about electronics, but also to share their knowledge and experiences. By means of a series of online

tools hacked together (the original Arduino website was made of two wikis, a blog, and a forum software), it was possible for hundreds of thousands of people to interact with one another. This was when we realized that we had something more in our hands: a platform of a kind, made of electronic circuits, software, online tools, and, crucially, a community of users who supported it in various ways.

When it came to my teaching, I jumped from spending days trying to develop original content to spending just a few hours extracting community-created content from the so-called Arduino Playground.⁹ A good example of this was Anders Gran and his classmates when they documented their first experiments with Arduino and Processing (Gran et al. 2005). When they started there were no freely available examples of how to do such a thing, but once they had done it and been published on Malmö University's website, they became an easy-to-use reference for anyone willing to teach that specific topic. Thanks to this and similar initiatives, I could teach my students how to become self-sufficient and search for documentation themselves. The growth of the community made the Arduino Playground an educational asset.

These blocks of educational content, when properly designed, can be reused in different contexts. For example, the content in learning how to control a motor can be used both when building a robot, but also in a more general context when learning about possible actuators. Information about a light sensor can be useful for a scientific experiment about the properties of light, but also when trying to detect the presence of a person in a room. A tool like Arduino, due to its open-endedness, can be reused in many different contexts. At the micro level, the same is true of the software running in the Arduino board, since the very same program can be used to drive entirely different sensors and therefore collect very different types of data from the world. On the macro scale, content about a certain topic can be empirically tested on this tool, and just a minute later

⁹ The Arduino Playground is a wiki on the Arduino website used by thousands of users to document new ways to interact with sensors, communicators, and actuators. As blogging waned and people focused on smaller-scale interactions such as Facebook or Twitter, the Playground became a more durable collection point and source of information in an open and self-regulated way.

a different kind of experiment can take place. Such is the power of digital technology. Yet in Arduino's success the main metric has been adoption—how people tapped its properties—the reusability of parts and content, the open-endedness of the tools, and ease of access to a vast quantity of resources—to make Arduino theirs, and how they then built their own toolboxes and kits from the knowledge gained by that experience. By enabling communication through a series of online systems, we helped create a community of users centred on the tools. The Arduino community.

This process saw Arduino transformed from a simple tool into a platform, with a set of kits and toolboxes designed both by us the developers, but also by the users. It was at this point that Arduino offered interesting case studies for my research. In the chapter on 'Lessons learned' I investigate Arduino as a situated, populated learning environment, and platform of sorts. While the effort put into creating and maintaining the Arduino community has left its mark on my practical experiments, my research has alternated between the various cases, from soundscape creation to haptic interfaces. The next section sets out the progression between the different cases, their relationship to one another, and the topic of this thesis, the creation of platforms as modular, transferable and generalizable systems.

Researching the common thread

Before helping Arduino become a successful platform for users at different levels, I spent some time on the IxD element in the research project 'Micromobility and Learning' (Cuartielles et al. 2003), exploring the possibility of using indoor positioning technologies and other meta-data to analyse how K3 students at Malmö University would relate to one another when working in a project-based learning environment.¹⁰ The results were possible technical developments and the seeds of future technologies: a collaborative sketchpad to allow students co-create sketches during brainstorming sessions from a multiplicity of devices (Cuartielles et al. 2004), a tool to design soundscapes where artists could enhance spaces by adding 3D sound

¹⁰ K3 is the acronym for Konst, Kultur och Kommunikation (Art, Culture and Communication), my home department at Malmö University.

thanks to a new extended markup format (Cuartielles & Malmborg 2004), and an indoor positioning system (Casas et al. 2002; Casas et al. 2004; Casas et al. 2007). While the prototypes and papers were of a highly technical nature, they had a clear pedagogical intention, as they were produced in the context of situated learning explored by the Micromobility and Learning project.

In one of my earlier literature surveys, I was fascinated by the work of the British artist Fiona Raby and her partner, the critical design specialist Anthony Dunne, at the Royal College of Arts (RCA) and by the people in their department, where the concept of critical design (Dunne 1999) was coined. Projects such as ‘The Presence Project’ (Dunne & Gaver 2001), ‘Flirt’ (Raby 2000) or ‘Edge Town’ (Hooker & Kitchen 2004) have influenced how I understand design as a mix of theory and praxis. Indeed, I still use the projects as teaching examples as I feel they in many ways anticipated our socio-technological present.¹¹ The reports and websites for these projects are classic examples of how to do research through design (see Chapter 4) using electronic objects as an interface between researchers and users. The objects themselves were not the outcome of the research project; they were vehicles for learning more about behaviour and interaction patterns. Aesthetics—as in the knowledge field of participatory and multisensory experience—played a significant role in all of the projects, as did the ambition to creatively and unconventionally explore the use of technology.¹²

From the same school of practice, and indeed in response to some of the early literature, William Gaver’s notion of cultural probes (Gaver et al. 1999) was one of the concepts that most influenced my first explorations in design. Given Gaver’s later revisions to his

11 For example, only the Flirt project anticipated the denial of service (DOS) attacks to mobile phone stations, geolocalized dating systems, or the value of simple animated images to convey meaning when using mobile communications.

12 While this thesis is not concerned with the aesthetics of platforms in particular, or of interactive digital systems in general, I should clarify my position on the aesthetic value of systems. As part of the School of Arts and Communication at Malmö University, I have come to believe that the aesthetics of digital systems (and by extension of life) do not rely on a traditional understanding of beauty, but on the sensual appeal of an artefact, process, or situation. Aesthetics, when viewed in this way, are strongly contextualized and not always generalizable. Thus, I could find myself exploring topics such as the ‘aesthetics of dirt’, the ‘aesthetics of interaction’ with a UI, or the ‘aesthetics of people flows in public spaces’. As the reader can imagine, this topic is far too broad to be explored here.

own method (Gaver et al. 2004) to alert designers to users and their lives, it should be noted that the ‘Cultural Probes’ study I conducted in 2003 was questionable in many ways. I created a cultural probes package designed to make usability tests for a technical device, the Anoto pen, using the very premises that Gaver et al. were to criticize in their revised method (2004). Cultural probes were designed to tap into an auto-ethnography of sorts from users that was otherwise difficult to reach, and the results of the analysis should be considered as nothing more than inspiration, suitable as a guide for the design process, despite my intention having been a quantitative analysis of the results of applying the method to a real-life scenario.

It was around that time that I drew up a manifesto for design involving users, electronic objects and everyday situations. The ‘Resign Desearch’ text (Cuartielles 2004)—which is included in the compilation here—set the tone for how I would operate from then on. I would focus on creating tools together with users, and through those interactions would create design collections. One example of this work was the series of joint projects with my students in 2004–2005, when we built an automated confession booth, a mobile phone with an address book hacked to react to the amount of alcohol in the user’s blood—stopping her from calling certain people when inebriated—and a chair that would first ask someone to sit on it and shortly after complain about the person’s weight. This collection of interactive pieces was only possible thanks to the students’ collaborations, and it helped me identify the way tools could be created so that students could build their own systems. By making the right choices when it came to the technology, students could not only implement the pieces by themselves, but they could contribute to the creation of a learning tool that would be reused by subsequent cohorts of students. Some of the pieces, being influenced by RCA’s critical design ideas, were intended to inspire users, illustrate a concept, or even highlight a weakness; others were conceived as explorations of future products and services.

In parallel, I was concluding an extended collaboration with the University of Zaragoza on the ‘Micromobility and Learning’ project. One of my demonstration kits, the Sound Space Development Kit

(Cuartielles & Malmborg 2004), had been created with reusability in mind. A platform for sound artists to create 4D sound experiences in a 3D space that was 10×10×10 metres, it used an indoor positioning system developed by the University of Zaragoza together with a software I designed for wirelessly connected handheld computers, and an XML format I defined (Cuartielles et al. 2003) to geolocate sounds in space and over time, it was a demonstration kit that I presented at a couple of academic meetings. The project had concluded by the time Arduino was created in 2005, having ended with a journal article (Casas et al. 2007).

One of the issues with digital technology at this point, as previously noted, was the lack of inexpensive and user-friendly tools for designers to start building prototypes. Arduino changed that landscape and my own modus operandi considerably. I had access to inexhaustible supply of the same reusable electronic brick to base my designs on,¹³ and there was a community that could furnish my need for certain snippets of code, and sometimes even anticipate it.¹⁴ And since every bit of Arduino technology was open source and built on multiple open-source components, it was possible to reuse it in multiple ways.

With Arduino taking off as a platform, I realized that it would be interesting to research how to apply some of the knowledge generated by that community to other cases. At the invitation of the Centro Cultural de España en México I embarked on a three-year co-design process to help relaunch a computer clubhouse for children. My contribution was to help establish processes, train a manager for the space, and create educational content, even co-designing a DIY robot with the help of a modified Arduino board, accommodated to the technology availability in Mexico City (Cuartielles & Yang 2010).

13 The term ‘brick’ is used extensively in the production of modular electronics. Hence the Lego programmable brick series, of which the RCX was the first. Originally created by MIT Media Lab, the Lego programmable bricks are microcontroller systems that can be reprogrammed from a computer for people to build their own artefacts.

14 In the open-source world, people produce code they need for themselves and then share it, but since it is early days yet, it is expected that the needs of one person will be the needs of multiple people at once. In that sense, some people can anticipate the needs of others—not that they can see the future, but their needs are similar for historical reasons.

The process took me further along the road of consciously reusing content, tools, and processes in other projects.

That was also the nature of my work done together with Stenslie, Göransson, and Olsson (Cuartielles et al. 2012a, 2012b, 2013; Olsson et al. 2012; Stenslie et al. 2014), included in this thesis. We collaborated on a series of prototypes to give physical feedback to users via haptics, and then to connect the users' bodies remotely using mobile technologies. The process of creating each prototype was a group effort, in which everyone had a different role but all joined in the 'heavy lifting' of construction, especially for prototypes with, for example, 120 interactive touch-points on the body. The projects advanced my research by requiring communication protocols and the creation of algorithms to allow hybrid sensor–actuator networks of hundreds of devices to transmit their state to the rest of the physically connected network, but also to others remotely.

My final block of research relevant here corresponded to my activities as head researcher for Arduino Verkstad AB, working on two EU research projects, SandS and PELARS.¹⁵ The process built on the idea of creating modular electronics, but this time by looking at the devices (sensors, communicators, actuators) with a view to making the 'smart' by designing an interconnection protocol that will allow for devices to be hot-plugged into the system and automatically negotiate their addresses. For the SandS project I co-designed a kit to help the researchers from the other participating institutions to connect any existing kitchen appliance to the SandS Internet cloud. This cloud would allow end users to instruct their kitchen appliances using instructions uploaded by other end users. In that way, if someone was to try to remove a strawberry juice stain from a white shirt, he would use his mobile device to interact with the SandS cloud, which following a semantic analysis would offer the best existing match for the task and then remotely control the appliance for the

¹⁵ Arduino Verkstad AB is the Swedish branch of the Arduino LLC company. It changed its name in 2018 to Arduino AB. SandS is the project 'Social and Smart: Social housekeeping through intercommunicating appliances and shared recipes merged in a pervasive web-services infrastructure' (grant 317947) which ran between 2012 and 2015 (<http://www.sands-project.eu/>). PELARS is the 'Practice-based Experiential Learning Analytics Research and Support' project (grant 619738), which ran between 2014 and 2017 (<http://www.pelars.eu/>).

user, giving it a set of instructions that we called recipes. The SandS project provided the first in a series of prototypes that were still not fully ‘smart’ in terms of how the system had to be implemented. The scenario of use included a technician who would come to the end user’s home to hack the existing appliances, combined with a computer application program so the various parts would interact with the cloud. PELARS built on the knowledge generated in the previous project to generate a more polished prototype ready for real-life experiments with relatively inexperienced users. In this case, a novice user would take boards corresponding to sensors or actuators and plug them together as a single communicator board using a bus configuration. Using a visual programming environment based on a flow paradigm, users could then program the relationships between devices, thus determining how devices exchanged information with one another and with other artefacts.

This, of course, was all part of the natural evolution of prototyping platforms. Platform designers today are creating modular systems much like the proof of concept we made for the PELARS project, designed to minimize the amount of time needed by users and developers to put together interactive systems composed of multiple sensors and actuators, system intelligence and communicators. I continue to study the relationships between users and connected systems, based at the Internet of Things and People Research Centre at Malmö University. The experience I have gained in building platforms—both from technical and user-centred perspectives—is in my opinion an asset when looking at how society sets about embracing the IoT paradigm. If it is true that everything that can be connected will be connected, as IoT promises, what are the implications? How will IoT be explained to people? How will we designers help developers communicate with users, and how will we involve all of them in building the IoT platforms of the future? How will IoT transform our understanding of society and how we relate to others? These are all questions that we as design researchers must address, and will be touched on as I explore IoT in this thesis. As the concept of IoT has taken shape parallel to the research presented in this volume, my

work in IoT is best understood as an exploration of the social and material premises of the field, materialized in the form of prototypes, communication protocols, file formats to represent physical properties, educational platforms, and wearable art installations.

Research questions

I work with demystifying technology and envisioning how to understand the future of ‘sensors, communicators, and actuators’ as Hayles (2009) interestingly chooses to label off-the-shelf smart devices. Such devices, with their ability to exchange information with other systems within a certain network, promise to be the building blocks in the creation of other artefacts. My research questions are thus whether this modularity, which we might refer to as transferability (in the sense that its affordances could be transferred to other designs) and generalizability (its properties could become a standard), constitutes a basic feature for a ‘platform-thing’—a sociotechnological construct for the running of reconfigurable applications—yet at the same time a sociomaterial artefact for collaborative learning? And given the definition of a ‘Thing’ by Björgvinsson et al. (2012) as a ‘socio-material assembly’—and my own definition of platform-thing—which kind of functional requirements can lead the design work towards the creation of a platform? In the context of this thesis, the questions have to be rephrased by asking whether the ability to reuse a system (or parts of it, taking advantage of its modularity) and reprogram it (changing its functionality without necessarily changing its materiality in order to accommodate one’s needs) could be basic properties of a platform.

In this thesis I explore this and a number of follow-on questions in depth. I present a series of cases (Chapter 3) in which I participated, and I will chart how the aspects highlighted in the theoretical definition of platforms (from Chapter 2) coincide with the properties of transferable and generalizable systems. I also respond to the questions in my literature analysis (Chapter 2), where I introduce the concepts, or rather preconceptions, which are transformed into a different collection of terms, as confirmed in the presentation of the cases and methods. The terms that will help designers when referring

to the outcome of their own projects are set out in ‘Lessons learned’ (Chapter 5), which traces the paradigm change in how to look at platforms that from the realization that humans are part of a larger assemblage that joins systems and people together. To articulate this, I argue that designing new technologies for learning (though it could be extended to other cases too) should allow for the better integration of humans in these assemblages, being more respectful of a variety of wishes and needs.

Limitations

The work presented in this thesis has to be contextualized within the scope of technologies for the creation of prototypes of devices, experiences, and services in the field of IxD. The text is a summary of thoughts, readings, and findings, and aims to provide guidelines on how to refer to the artefacts and software produced during a design process, both during and at the end. Far from being a purely theoretical body of work, my research has been conducted by direct involvement in the co-creation of systems as part of or in collaboration with various teams of people over extended periods. Sometimes it took years to iterate concepts and prototypes due to the nature of the projects themselves, the funding available, or—when in the lack of funding—people’s availability to work for free within a project.

My role in different projects has varied. In some cases, I designed the technology, in others I facilitated the community while doing system maintenance, and I even became the main designer for EU-funded projects. This multidisciplinary approach both within the teams and around my research persona required that I adopt a range of strategies in order to cope with design situations. I had to develop an understanding of how to behave in different cases and how to get projects off the ground. But it was not always possible to do everything as I would have liked, being a team player-cum-activist, in the sense of Hale’s activist research manifesto (2001).

Even though I have worked with hundreds of stakeholders over the years, I did not focus on the ethical aspects that now engage contemporary thinkers. Privacy and data sharing were not my primary

concern when working on the creation of connected home appliances for the SandS project. Not that I am unaware of the potential downsides of new technologies such as always connected, always-on devices, but it was not my duty in the scope of that project. The same applied to the PELARS project, where we built interactive prototyping platforms to gather data about how students interacted with one another and to send reports to both students and teachers. There was an ethics group within the project, but I was not directly involved in its work.

Nevertheless, it is worth dwelling on the shift from a problem-based toolbox approach to a community of learning as embodied by platforms. Toolboxes belong to the realm of experts, as I will explain in Chapter 2, and are grown from the individual experience of a person navigating the field. On the other hand, communities are built through the interaction of many people and can have different governance models, which in turn may determine their agendas for thought and action. While the expert responds to the rules of a field, under a self-determined governance model communities can easily drift. This has certain ethical ramifications that must be taken into account, not the least the central question of accountability.

The question of accountability, most acute in the concept of assemblage (Latour 1996, 374) and the involvement of non-humans, is a source of ethical conflict that has yet to be resolved. Can a non-human (such as a program in a computer) or a whole assemblage (a platform) be held accountable for its actions, or should the human creator/s of that entity be wholly responsible for its actions? In the case of a technologically sustained community, how should accountability be shared? To what extent should non-humans be allowed to impinge on community governance models, and to what extent should humans be allowed to experiment with technological forms of power? These and similar questions will be addressed with the help of the literature. While Jane Bennett and others have shown that the control of everyday life is de facto dependent upon fragile assemblages of humans and non-humans (Bennett 2005), this will not stop us from creating new exploratory communities of learning to deal with technology, and gives rise to even more questions. Can

a technology-driven education free us from a society mediated by (and potentially controlled by) technology? Is technology the cure for technology? While none of my projects or articles have dealt directly with issues of accountability, they remain significant ethical questions to take into account and, by democratizing technologies, arrive at alternative approaches to scaled-up infrastructures such as the Internet of things, which are of immediate concern in this thesis.

Finally, there were all sorts of technological shortcuts we had to take in the creation of many of the technologies described in this dissertation. These technical limitations had to do with the nature of the prototypes, and the difficulty of anticipating how well technology will scale up in experimental settings. For example, the European research project SandS was tested at different research facilities, but was never put in hands of end users. In this kind of research, there is often clear limitations on the possible outreach of technology besides the theoretical calculations made for the project. In other projects, such as the ‘Creative Technologies in the Classroom’ project (see Chapter 4), in which over 17,000 users participated, it was evident that the technical decisions we made could be scaled-up. Instead, I have to ask myself from an activist perspective whether scaled-up is enough for successful outreach. From a research perspective, and especially design research, having an experiment tested with a certain degree of success on tens of thousands of people is an obvious success, but is the proposed design good enough to reach beyond those tests? Is the concept itself durable? In the context of technology, which is constantly evolving, how durable can projects be? I provide readers with the tools to answer at least some of these questions in subsequent chapters.

Summary

I came to IxD with a mixed background in project management and microelectronics design. My first task was to create the educational content for programming for IxD undergraduates. It was my understanding that the students participating in my courses would have no knowledge of the basics that I had been introduced to when studying

engineering. Like every course taught for the first time, I spent most of it trying to convey a message that could be understood by the students given their previous knowledge in the field.

The process of teaching was (and still is) a project in itself: when any of my courses start, I have no clear idea of the outcome. Students define their needs, and through their interaction with me and other lecturers they co-create new concepts and prototypes for interaction patterns, experiences, complex systems of products and services, and of course electronic devices. This is the context for my research. I spend my days interacting with others, looking for effective and sustainable ways to develop and iterate prototypes made of software, hardware, or both.

Over my years of work on the creation of tools, kits, and platforms, I have been active in both theoretical and practical research on a whole series of initiatives. I unknowingly followed the steps described by Gaver in his 2012 analysis of research through design by beginning with a manifesto, in which I set out the grounds for my methodology, with collections instead of single artefacts as project outcomes, and defining how I would relate to users doing the sort of activist research defined by Hale (2001), where I was conducting user-oriented research into cases that seemed worthwhile because they would enable access to technology in novel ways for different groups of people, whether students in Malmö, teachers in Spain, researchers in Italy, or children in Mexico.¹⁶

Parallel to the process of defining my working process, I co-developed as part of the ‘Micromobility and Learning’ project (Casas et al. 2004) an indoor location system and various applications to demonstrate its functionality. Probably the most remarkable was a system to design soundscapes that used the positioning technology to place sounds in space, with installations which users experienced via handheld terminals (Cuartielles et al. 2003). My results gained me a couple of research residencies, one of them at the IDII, where I met the people

¹⁶ See Chapter 4 for the full definitions of activist research, which, according to Hale 2001 aims at developing a third category of research, a hybrid one that should be called ‘user-oriented basic research’.

who would join me in co-creating the Arduino platform. I spent several years running Arduino's web platform, which left me with a much clearer understanding of what people were interested in when looking at prototyping tools applied in different contexts. I became very active in the creation of educational experiments, initially for universities and later for schools, and moved into the creation of full educational programmes that were deployed in several regions in Spain, Sweden, and Ecuador.

While developing and testing educational programmes, I continued to research the creation of wearable technologies by looking at simple interactive systems that could be deployed in a variety of sizes, and which would get people to connect to one another through the use of touch and haptic feedback as an interface for remote communication. The work done with Stenslie, Olsson, and Göransson (Cuartielles et al. 2012a) in the field of haptics led me to think about new interactive platforms where it would be easy to build interactive systems using a mix of modular off-the-shelf electronics and visual programming languages. The first step was the implementation of a new communication protocol, along with a new flow-paradigm-based visual programming language for the design of the taxonomy of multi-processor systems. The two EU research projects I led for Arduino Verkstad AB between 2012 and 2017 were iterations of these ideas of creating expandable, multiprocessor, hot-plug systems.

While the work on haptics looked at the creation of identical cells that would interconnect people remotely by touch and vibration, and the EU research projects explored the possibility of anyone being able to build similar systems but in a much more open-ended way, my current research on the Internet of Things and People (IOTAP) as part of the eponymous group at Malmö University is aimed at building a more critical discourse on the way the IoT is conceptualized and realized. My experience in building tools and platforms means I approach the field with a critical eye. To reach a new distributed computing paradigm like the IoT is going to require the application of platform design thinking, and users and developers will have to collaborate in making it happen.

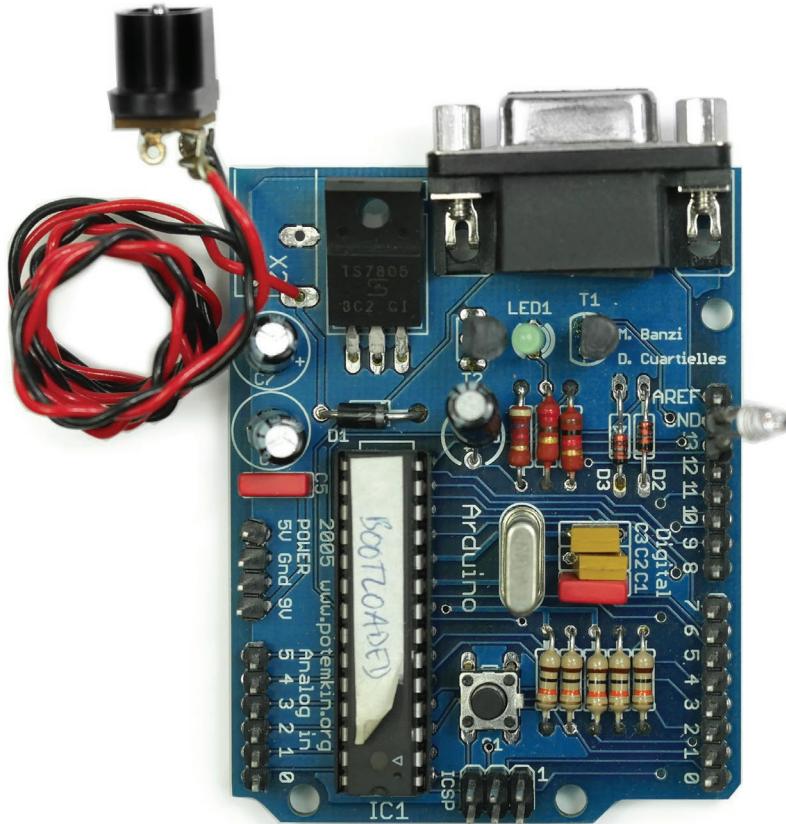


FIGURE 2
ARDUINO SERIAL BOARD V1, 2005

2 THEORETICAL FRAMEWORK

This chapter deals with the theoretical framework developed to support my investigation on platforms. Mine is a composite body of inquiry conducted through active participation in building digital systems over a period of some fifteen years. Some of those systems, which I came to call platforms, have become successful, while some others vanished almost as quickly as they were built. The following sections introduce a series of concepts that clarify the underlying premises of my research during the development of the projects that are presented in this compilation thesis.

The chapter looks first at observations and then preconceptions. The section on observations frames the literature used to build an understanding of platforms, and the ethnographic and design tool-boxes they require. The section on preconceptions introduces the key concepts that will help readers navigate the projects and understand the level of finish and participation each of the cases has to offer. Thus while the observations section looks at different theoretical views instrumental in building an understanding of contemporary complex systems such as platforms, the preconceptions section explores and discusses my own framework. Evolving under the premise of incompleteness, long after the publication of the various papers, this preconceived framework will be articulated, challenged, and transformed.

From networks to platforms

From a purely technical perspective one could claim that digital networks are nothing but a series of computers (clients and servers), exchanging information (via protocols) and storing the data (in databases) for later use. This is however a very narrow definition of how to look at contemporary systems. Clients are now pervasive, while servers have been diluted with cloud storage. It is not only the

technology that has changed from a series of discrete artefacts into a sort of continuum. The term ‘pervasive’ infers that technology is always there, always-on, in our pockets, usually in the shape of a mobile phone. Technology is no more a ‘window on a virtual world’ than it is an axiom of the cloud’s brain that connects us permanently to a database.

People are now part of the continuum. We have redefined our role and become one with the ‘distributed machine’ that the network has become in recent years. We no longer interact with others using the system as a vehicle, because the affordances (Gibson 1986, ch. 8; Gaver 1991; Fry & Reas 2007, 554; Kaptelinin & Nardi 2012) of the artefact allow for better interaction and integration.¹⁷ Rather, we interact as part of the system—a regulated part that we often call a ‘platform’, the underlying network of which is now diluted in the cloud.

In order to avoid a discourse determined by dichotomies, in the following genealogy I instead emphasize the historical transformation of terminologies and understandings of the relevance of technology in the field of platform creation. It is my goal to draw attention to a modality of design rather than to oppositions in the existing ways of making technologies for people.

From accessibility to democratization

The increased accessibility of communication infrastructures such as telephony or the Internet is commonly referred to as ‘democratization’, a process that Alexander et al. (1977, 73) studied through the decentralization of the power in communities, while Ehn (1988, 247) has looked at the more concrete example of the workspace with the introduction of ‘computer artefacts’. The word democracy, from the Greek *demo* and *kratos*, ‘may refer to the power (*kratos*: literally, “grip”) of the *demos* in the sense of its capacity to act’ (Lane 2014,

¹⁷ This way of understanding affordance extends this physical property to screen-based artefacts. In Fry & Reas 2007 it is briefly mentioned in Golan Levin’s additions: ‘Some of the most popular machine vision toolkits take the form of plug-ins ... Such plug-ins simplify the developer’s problem of connecting the results of the vision-based analysis to the audio, visual, and textual affordances generally provided by such authoring systems’ (554).

55). *Demos* may both refer to ‘the people’ and ‘the common people’ (‘the many’ or ‘the crowd’). As Lane (2014, 87) mentions, it is an ambiguous term, which is as much a weakness as it is a strength. This duality leaves democracy a term that could refer to giving equal power and rights to all levels of society, but equally to the power that the many can exercise over the few. Democratization is thus a term I use to talk about giving different levels of access to the public.

The relationship between technology and democracy can be seen from different perspectives. On the one hand, better access to technology increases the social debate and the participation of individuals in governance (Sun & Barnett 1994), this means that there is a greater chance for a democratic process to emerge in highly connected societies. While Sun and Barnett’s study was limited to fixed telephony communication networks,¹⁸ others such as Cubitt (2014) mention that data network developments are important to democratic participation. It could be argued that if this bond between technology and the democratic process gives more people greater access to fundamental communication technologies, it is an act of democracy. Not only is access relevant, so is the possibility that certain technologies offer for knowledge creation. Open-source and free software, together with the more recent hardware hacking techniques and the open hardware licence models, establish a ‘new set of opportunities for democratization of knowledge’ and ‘develop new forms of technological citizenship’ that even affect the production of industrialized products (Powell 2012).

We can find a counterargument in the work of the philosopher Andrew Feenberg on democratizing technology (2001). From a neo-Marxist perspective, Feenberg defends the view that we should find a way to rationalize technological development so that it can integrate alternative societal views and values. His early work on the topic might lead the reader to believe he is against the idea that spreading technology as much as possible is an act of democracy; however, in later publications his point of view is clearer. Thus in his 2010

¹⁸ Sun & Barnett 1994 suggested that other networks, such as computer networks—the Internet—or the satellite network, should be investigated to gain a better understanding of how their development was influencing the spread of democracy.

book *Democratic Rationalization* he argues that the development and impact of technology are ‘intrinsically social’ and that democracy has to be extended into the ‘technically mediated domains of social life’ for it not to disappear (Feenberg 2010, 6).

Democratization, filtered through Feenberg’s discourse, is linked to accessibility, which refers to making the technology present in the contexts where it is needed. Empirical data suggests that accessibility could be increased by letting users own the infrastructure, if only partly, through hybrid ownership and exploitation schemas (Baig et al. 2016; Saldana et al. 2016). While accessibility deals with ownership, democratization refers to the user’s rights to influence the creation of the technology per se. Contemporary discourses on design and technology suggest that our approach to democratization should see the making of horizontally governed communities of limited size (Alexander et al. 1977; Papanek 1988).

The axis of analysis for accessibility and democratization raises an interesting conflict that platform designers will have to solve: a distributed community will have a harder time accessing economies of scale, as its power of negotiation in the current market economy will be reduced. Which balance can be struck between community size and technological scale that would generate an economically sustainable platform? On the other hand, platforms exist that do not operate with this distributed paradigm. How can ideas of accessibility and democratization be operationalized on existing online platforms such as Google or Facebook, where the number of users is what matters? Expanding the question to a global scale, is a distributed architecture really a viable solution for a fully networked society?

From distributed networks to interdependent actors

A different take on democratization comes from Saldana et al. (2016). As engineers, their take on the democratization of access to different types of networks is to open up for the creation of a whole new range of services based on alternative economic and ownership models. The most relevant aspect here of the Saldana study of technologies used in remote locations or places where it is not financially worthwhile for large corporations to offer access to digital data networks, is

how a variety of stakeholders will intervene and share the purchase, deployment, maintenance, and exploitation of different networks. Furthermore, such deployments allow for different business models as well as shared ownership mechanisms that distribute the power of the networks differently. When looking at a deployed technology in this way, we give it an agency similar to human agency. This shift in the understanding of agency is often referred to as the ‘material turn’, ‘new-materialism’, or the ‘ontological turn’, depending on the school of thought (Pellizzoni 2015, 72). The ‘material turn’ refers to the analysis of complex assemblages of non-humans and humans, where the latter are not privileged in relational terms.¹⁹ This combination of different types of actors, their relations with one another, and the maps of connections traceable by researchers when describing them, is also the central idea in actor–network theory as coined by Bruno Latour and others (Law 1992; Latour 1996, 2006). Actor–network theory has been applied to a wide range of different cases where technologies—as non-human actors—share agency with humans.

In an article on ‘The agency of assemblages and the North American blackout’, the political scientist Jane Bennett (2005) looks at systems as an assembly of operators (humans) and machines (non-humans), where the decision-making is distributed among the two groups of stakeholders. According to Bennett, a network is an assembly of hubs, switches, servers, but also IT technicians, users, and service providers. The governance of the network is controlled by a collective intelligence, in which machines as well as humans make decisions at the speed of light. The networks analysed by Saldana et al. (2016) have a type of agency, due to the different nature of the participants in the everyday management of the systems, that differs from more mainstream networks. Saldana et al. suggest that people—understood in a very broad sense to include end users, small-scale entrepreneurs, and local authorities—not linked to large telcos can make use of state-of-the-art technologies to create their own communication

¹⁹ It is even possible to find an early reference to such an understanding of society in the physicist Steve J. Heims’s introduction to Wiener’s fifties classic, *The Human Use of Human Being*: ‘the long-standing mind–brain duality was overcome by a materialism which encompassed organization, messages and information in addition to stuff and matter’ Wiener (1989 [1950], xx).

infrastructure. Off-the-shelf, commercially available technologies are good enough to support the creation of new situations where accessing online products and services is somehow more democratic.

Seen from Bennett's perspective, contemporary systems are technosocial and/or sociomaterial, not only made of wires and bits, but of flesh and blood and social exchanges. Automated mechanisms are interwoven with humans in complex feedback loops where decisions are made on the spot on the basis of information coming from mixed media systems. Her study of the North American blackout in 2003 reflects upon the issues of decision-making, human-machine assemblies, and spatial-temporal changes. When applied to the idea of building a network in an underprivileged area, the concern of Saldana et al., the social aspect takes on an even more complex dimension, where local communities, represented by political parties, could introduce socioeconomic agendas to the governance of systems that are already commanded by technology and the technological culture.

From a technical perspective, the ethnographic data captured by Saldana et al. (2016) shows how the Internet is indeed a network of networks. In that sense, we find ourselves faced with a distributed technical network with different models for sustainability and governance depending on the case. From an actor-network theory perspective, what we see is a series of relationships between actors: users, entrepreneurs, networking machinery, the electricity network, and even the geographical landmarks that determine the distributed nature of the technical network. Actor-network theory becomes a tool with which to trace complex situations in which many interrelated actors coexist, who by virtue of their agency all contribute to make the actor-network what it is.

From embedded systems to the IoT embodied

The literary scholar N. Katherine Hayles (2009) goes beyond Bennett's global view of systems in her analysis of radio-frequency identification (RFID) technology as the stepping stone to largest platform-to-be of all: the Internet of things. RFID is a system based on tags that can be embedded into almost anything and readers that can identify the tags at a distance by sending a radio signal. RFID tags provide

a way to equip things with unique identifiers—a sort of passport number—and this renders a new kind of agency to things. It is no more *a thing*, it is *the thing* we are talking about. Unique identification is part of human nature (though not only ours, as animals also have it) that we can now transfer to inanimate objects. For Hayles, a thing is a smart device with some sort of distributed computing capabilities. RFID is distributive by nature and makes objects into things. This brings with it a transformation of our understanding of the world: we are moving away from an understanding built on the triad of human–animal–machine to one filtered through the lens of a new triad of human–animal–thing. Hayles’s definition of thing is in this respect of central importance. Things are a new type of object with double-layered properties, for they have a physical reality but also a virtual one, and both do not always stick together.²⁰ Things are made of ‘sensors, communicators, and actuators’ (Hayles 2009, 57), in which the communicators refer to the ‘new’ capability that things have to talk to other things within networked systems. Communicator is thus an interesting label when referring to an off-the-shelf smart device that will give off-the-shelf sensors and actuators the ability to exchange information with other sensor–actuator systems within the same network.

The promise of Internet-working, communicating set-ups does not stop there. Using this technology, devices connect to other devices outside their networks of origin; data is collected by third parties; and information is extracted from patterns and events, to be harvested from data flows (Brody & Pureswaran 2014). The embedded sensor–actuator systems—electronically enhanced apparatuses that could range from a wearable computer in our pocket to a weather monitoring network covering a whole region—when augmented with Hayles’ communicators, constitute the things in the Internet of things. While the power of embedded systems is such that they are always on, now they will always be connected too. In the mid 2000s, the legal scholar Lawrence Lessig (2004, 297) fantasized about

20 Note the difference between Hayles’ ‘thing’ and Björgvinsson et al.’s ‘Thing’. It is not only in the notation, but also in the meaning. For Hayles, a ‘thing’ is a smart device with some sort of distributed computing capabilities. For Björgvinsson et al., a ‘Thing’ could be anything accounting for the result of a co-design process.

having always-on systems that were always connected. Pushing it even further, the science fiction author Bruce Sterling coined ‘SPIME’ to describe manufactured objects with an informational layer so rich that they would be referred to as ‘material instantiations of an immaterial system’ (2005, 11). Less than 15 years later, with the IoT being implemented in live scenarios, always-on, characterized by layers of meta-data systems, is already spreading all over the world. This is a new reality, a distributed material turn where anything and anyone can have an always-on virtual representation in the form of Hayles’s human–animal–thing triad.²¹ The two predominant computing paradigms of pervasiveness (always with you) and ubiquity (embedded everywhere) have gained a new definition—embodied everywhere. Technology is embedded in the body; the thing becomes one with the living being. According to Redström (2001, 213), take the concept of ubiquitous computing to the extreme and everything we do can be considered writing (as in writing data to the system). When carrying around IoT-related technologies, we are writing all the time, when we move, when we speak, or when we touch other things.

Looking deep into this conglomerate of flesh and bits, wires and social exchanges, we can ask ourselves if it is this hybrid nature that makes a platform. Is IoT a platform? If so, what is the role that humans will play in it? Who owns the platform—the users and their devices, or the developers and their services? How does authorship work on a platform, and who owns whatever is produced on it? Some of these questions can be answered by further exploring the existing literature on the topic, while some others can only be given more speculative answers as they are still unknown. I will start by exploring the meaning of the word platform.

21 Note that in contemporary engineering literature we talk of virtual sensors, so these do not necessarily represent one-to-one a physical sensor in virtual infrastructure, but more groupings of physical sensors representing virtual properties. For example, presence can be estimated by measuring the number of times a door is opened, added to the image from a camera and infrared light reflected into a passive IR sensor (a movement detector). Since the virtual sensor is always-on infrastructure, it does not matter whether there is a real sensor; it can always compute its data from pre-existing records, or even infer the information from other sensors on the periphery.

From technological switches to intermediary exchange

The term ‘platform’ has been used broadly as part of a discursive work, mainly because of the different connotations of the word, to accommodate all sorts of points of view. A platform can be understood as something technical, as a place to address an audience from or as a moment of opportunity. It even has a figurative connotation, becoming a metaphysical space ‘for opportunity, action, and insight’ (Gillespie 2010). It is Gillespie who offers the best description of what a platform has become in contemporary history: ‘The term “platform” has emerged recently as an increasingly familiar term in the description of the online services of content intermediaries, both in their self-characterizations and in the broader public discourse of users, the press and commentators.’ Platforms grow on top of networks, and more specifically data networks that take advantage of online content brokers. Gillespie studies YouTube, a system that does not create content itself, but collects content created by users, and curated by means of machine-controlled algorithms characterized by user preferences.

The term platform is both specific enough to mean something and vague enough to work across multiple domains. Gillespie groups the different meanings of the word into four categories: computational, architectural, figurative, and political. As a computational term, it is ‘an infrastructure that supports the design and use of particular applications’. As an architectural term, it ‘describes human-built or naturally formed physical structures’—it seems to be the combination of the French terms ‘platte’ and ‘fourme’ which translates as ‘flat form’. As a figurative term, a platform becomes a ‘metaphysical [material] for opportunity, action and insight’. As a political term it ‘generally implied a kind of neutrality towards use—“platforms” are typically flat, featureless and open to all.’

To that common set of features, I would add the idea of community, defined as a ‘realm of practice’ built on ‘connections uncovered in the course of everyday experience’ (Feenberg 2007, 28). Platforms as emergent systems can be born from a social innovation process, where a community would foster the creation of a system to suit their

needs, but where there could also be a purely economic interest in their creation as suggested by Plantin et al. (2016) in their analysis of Google and Facebook. This idea of a dual origin of platforms can also be found in Saldana et al. (2016) when looking at the different types of socioeconomic networks that can be formed on top of a technical communication network: some of their cases are bottom-up (more community-centric) while others are just clever adaptations of business models of more traditional corporate systems to accommodate a situation or need. Gillespie (2010, 350) hints at the idea of community when he explains how a platform ‘suggests a progressive and egalitarian arrangement, promising to support those who stand upon it’. He takes this line even further by analysing what makes computational systems become computational platforms. Specifically, he says that a computational platform is the technical base ‘upon [which] other programs will run’; but what makes them platforms is not that ‘they allow code to be written or run, but because they afford an opportunity to communicate, interact, or sell’. This community aspect is then linked to the idea of the platform as a political term. Communities are governed by social contracts (terms and conditions or codes of conduct) and so is a political platform; what it takes is a set of beliefs to build on (Gillespie 2010).

Again, Gillespie explores in depth some of the relevant attributes of platforms, of which the ‘featureless’ property of a platform seems to be one of its strengths: a platform ‘is anticipatory, but not causal ... implies a neutrality with regards to the activity’ (2010, 350). This statement raises a series of questions regarding the relationship between neutrality and features. Do not platforms that cover the same design space compete in terms of features (whether technical or any other form) in order to survive? Is that neutrality just a temporary aspect, seen only during the establishment of a platform? To what extent is a platform such as Facebook or Google neutral? Is there a difference between large platforms and small ones?

The question of neutrality is addressed by Feenberg (2010, 6) when he states that ‘modern forms of hegemony are based on a specific type of technical mediation’. In the current state of affairs, technology serves specific interests rather than the interests of ‘people’, and

therefore fails to support democracy unless a radical change is effected. Ultimately, Feenberg raises the question of whether technology, by definition, can be democratic. He suggests that unless there is a change in how technology is pursued, democracy will simply fade away. It is a catch-22 situation, as in order to support democracy we need a more democratic way of creating technologies. He concludes with a message of hope, though: ‘Technology can deliver more than one type of technological civilization. We have not yet exhausted its democratic potential’ (Feenberg 2010, 29). Perhaps, as Powell puts it (2012), ‘hardware hacking’ with its ability to influence the production of industrial products, can be the radical move that Feenberg hints at to make technology more democratic.

From program to accountability

For software developer and entrepreneur Marc Andreessen, known as the co-author of the Mosaic Web browser among other things, a platform is something that can be programmed, and if it cannot then it should not be called a platform. His definition comes very close to Gillespie’s ‘computational platform’, as we have seen. On the other hand, if one reads Andreessen’s 2007 article ‘The three kinds of platforms you meet on the Internet’ and follows his argument in depth, it is evident he has a clear vision of what a computational platform should be in terms of its offering for—mainly—developers. But then it is a statement made by a person devoting his time to the creation of online programming platforms. This idea of programmability as the basic property of platforms has permeated a more scholarly discourse too. For example, the Dutch researcher Anne Helmond, when looking at the infrastructure of social media platforms, describes ‘platformatization’ as the introduction of programming capabilities on the web, and more specifically the ‘programmability of social media platforms for the web’ (Helmond 2015).

Andreessen (2007) represents a far more technocentric discourse, as he is obviously addressing an entirely different audience than the authors considered thus far. He is worth mentioning, though, to draw attention to the fact that there is a whole group of potential stakeholders who have a clear idea of what constitutes a platform and what they expect from a platform design process. For designers,

it is important to figure out how to approach the needs of this kind of user, who, following Andreessen, tend not to waste words on the ethical aspects, which according to Feenberg (2001) we should always take into account when dealing with the design of new technologies.

Contemporary research efforts under the aegis of the European Union are already looking at how systems should be designed to take ethical aspects into account. For example, the EU research project ‘Virt-EU’ is looking at the issue of citizen rights in the management of personal data. At an official project event at the IT University in Copenhagen on 12 January 2017 (Virt-EU Kick Off 2017), Alison Powell and Alessandro Mantelero introduced the audience to the need to include ethical considerations in the design of IoT systems.²² Powell discussed our position as citizens in this seismic sociotechnical shift by saying that ‘technologies of datafication, that are transforming our life more and more into data, create new dynamics and new relationships that can affect the way we are citizens ... A good citizen per these processes, is the one that produces a lot of data’ (Virt-EU Kick Off 2017). There is room for people dealing with the idea of datafication of our citizenship. And here is where the ethics come in. Powell advocates the re-examination of the principles of ‘virtue ethics’. She thinks that they will come from the people who are deploying the technologies. In her Copenhagen speech, Powell introduced the term ‘virtuist’, a person who proposes a series of virtues so that people flourish, and will be the bedrock of a good society.

Mantelero, both at the IT University event (Virt-EU Kick Off 2017) and in an earlier paper (2016), focused instead on the legal aspects of data acquisition, treatment, and protection. At the heart of his Copenhagen presentation was the assessment of risk. In general terms, it is an element that has been present since the advent of data protection regulation. It is based on the idea that data collection created an awareness of the risk of social surveillance, and therefore

²² Virt-EU (<https://virt-eu.nexacenter.org/> [11 Feb. 2017]) is an H2020-funded EU project. During the Virt-EU launch, Alison Powell gave a talk on ‘Dilemmas of Connected Experience: A Virtue Ethics Response’. Alessandro Mantelero gave a speech titled ‘The ethics of the Internet of things—what kind of future do we want to live in? The EU General Data Protection Regulation and IoT. Legal issues of the risk-based approach.’

the regulatory bodies had to put in place protection for individuals. His goal is to study the IoT communities that exist and look at whether the communities' values can be applied to the law and vice versa.

In her essay 'Deadly Algorithms' (2014), Susan Schuppli addresses the concept of algorithmic accountability, and explains how 'decision-making by automated systems will produce new relations of power for which we have as yet inadequate legal frameworks or modes of political resistance'. She does not seem optimistic about our chances of identifying valid governance models that will tackle the issue of sociotechnical assemblies making potentially deadly decisions for us. Schuppli infers that society has to undergo a significant transformation in order both to enable these technologies and to establish mechanisms of control, and that will require of a greater level of participation. She asks an open question about the kind of social assembly capable of controlling these advanced technological developments, especially in situations where the information itself may be kept from the public for 'reasons of national security'. Powell and Mantelero, in looking at sociomaterial assemblies, are not only saying that we the public become data that feeds a series of supranational intermediaries that profit from that data, but also that associations of people—communities—are currently unprotected as a group. While individual rights seem to be covered by contemporary legal frameworks, group rights are not.²³ By extension, sociomaterial assemblies are not covered by the same regulations that protect individuals. Powell and Mantelero's thesis is that we should support the autoregulation of assemblies by including the consideration of legal values when designing platforms. Their work is more recent than Schuppli's, and they seem to be looking for solutions to the problems she describes.

From usability to participation

Björgvinsson et al. (2012) define 'Things' by presenting the term in contrast to the traditional industrial design goal of the production

²³ My intention in mentioning this here is not to start a conversation about legal frameworks and the like, but to highlight that socio-material assemblies are hard to regulate, and that their form changes by doing something as simple as jumping from the individual to the collective.

of objects. The ‘design community [has to] move from designing “things” (objects) to designing Things (socio-material assemblies)’ (102). In order to construct their own view on the topic, they find that the ‘etymology of the English word “thing” reveals a journey from the meaning of a social and political assembly, taking place at a certain time and at a certain place, to a meaning of an object, an entity of matter’ (102). In a way, the concept of Things reclaims the social meaning of things, creating a new super-term that includes both the old and new etymologies in one. This redefinition of the term should be contextualized within the general discourse of how designers should engage in co-creation processes, addressing societal challenges through the hands-on iterative processes where prototypes are created. Their move from things to Things is directly connected to the idea of having to move from designing tools to designing platforms. Designers, they argue, should consider the sustainability of the design and think beyond the specific project ‘toward future stakeholders as designers’ (Björgvinsson et al. 2012, 102). This they define as ‘infrastructuring’: a design process that focuses on the building blocks conforming a new complex system, understood as a amalgamation of products and services for stakeholders to put together. Infrastructuring has a strong element of participation in what is seen as a situated and continuous design process, hence the reference to the cultural formation of Things. The question is whether designers can follow this approach, and whether there are any methodological tools to encourage or facilitate participation.

One possible approach would be to use ‘sustainable’ human–computer interaction as presented by Raghavan and Pargman (2017). Their suggestion is to use ‘disintermediation’—the redesign of systems to remove intermediaries—as a way to simplify access to products and services, but also to simply make platforms cheaper. It has to be understood that not all intermediaries in assemblages are bad. The ‘punctualizations’—or process of representing a whole network of resources with a single node—as described by Law (1992) can be understood as adding an intermediary between the actor and the resources, offering simplicity in exchange for control.²⁴ An example

²⁴ Law 1992: ‘punctualized resources offer a way of drawing quickly on the networks of the social without having to deal with endless complexity’.

of such disintermediation is the case described by Law and Mol (2001) of the Zimbabwe bush pump, where the design of the pump was left open so people could use it in whatever form they considered convenient. In that case, the disintermediation is clear in the fact that anyone could make their own pump from a generic model, which with a little tweaking could be adapted to meet any demand; it would not be easier to use, but easier to ‘hack’. By contrast, Akrich (1992) presents the case of solar-powered lamps, where a complex system was ‘punctualized’ into what at first seemed to be a very simple technical solution. Yet it turned out to be simple only at a superficial level. Due to the specific punctualization, in this case the kind of connectors between lamp and battery and the fixed length of cables, the lamp was hard to install and not always usable. Here, the attempt to prevent users from accidentally damaging the equipment or themselves by the use of standardized-yet-proprietary designs made it hard for local electricians to install and maintain the technology. This is the opposite of a disintermediated process—an entity designing for a user group failed to produce the exact design needed, and since it was not possible to adapt it in the field, the design did not work as wanted.

These two examples speak to design accessibility, agency, and the pervasiveness of technology. Infrastructuring as a process of ‘aligning socio-material public Things’ (Björgvinsson et al. 2012, 108) is about enabling participation in design, and that is how this concept of designing things links to platform design in a seamless way. The featureless, generic character of platforms, which Gillespie (2010) talks about, is of central importance in order to facilitate a sufficiently open and continuous design process. For the newly created system to be a thing, it cannot be too specific. Yet if pushed too far, its generic qualities may become standard, with little or no room for local decision-making or adaptation.

One method of addressing the potential of a democratic process on platforms could be the implementation of feedback mechanisms by design to allow continuous participation. Platforms, because of their technological component, are able to implement feedback loops. Wiener (1989 [1950], 61) describes feedback as a ‘method for controlling a system by reinserting into it the results of its past per-

formance'; it is a 'complicated process of discrimination, regulated by the central control as a logical or mathematical system.' As a matter of fact, Wiener describes 'policy feedback' as the type of feedback that will produce either a conditioned reflex or a learning process (1989 [1950], 33). Feedback is an important part of infrastructuring, as is learning. Wiener's ideas are exportable to any kind of (non-) human assemblage, and therefore contribute to the idea of continuous change through participation that should command the creation of things. Participation is therefore not important only during a system's co-creation phase, but also once the platform has been deployed. The possibility of having meta-conversations on platforms where even the nature of the platform is questioned opens up for the creation of codes of ethics and community governance models, which might challenge the original intentions of the platform builders as to how discourse should be handled. This application of the cybernetics concept of the feedback loop raises other interesting questions once agency is passed to the non-human component of the assemblage. If, as Rouvroy (2013) said of algorithmic governmentality, artificial intelligence is constantly running on feedback loops that modify non-human responses in real time, how would non-human feedback loops be weighed against the human ones? They would presumably challenge some of the human roles in the assemblage, such as the ability to curate or highlight aspects that might be considered of greater relevance to users. Boden (2017, 31) suggests introducing heuristics by design to the assemblage so that the system anticipates in which cases the human or the non-human should count first in the event of a conflict. Such systems would affect participation and governance, potentially shifting them from being discourse-driven to being purely data-driven. Some of these challenges will be explored later, while others are part of the current discourse of the agency of actor-networks, and thus are already detailed in the literature.

Preconceptions

Preconceptions, the starting point of the present study of platforms, are germinal ideas that will later be challenged and augmented by experience. Initially, I took technological development to be a process of zooming out from the micro scale of a tool to the macro scale of

the role played by that tool as something bigger and generalizable.²⁵ I started with the idea that tools could grow organically into platforms through the addition of other processes and tools. One of the cases I am still working with, the co-creation in 2005 of the Arduino platform, encouraged me to believe that platforms could be constructed slowly and sustained easily. This preconceived idea of how platform creation could be was based on a single experience, and therefore detached from other realities. Dealing with platforms today, in the context of the market economy, demands that one always look at the metrics, constantly checking how well one's platform performs in comparison to others in the same sphere. It is an exercise in numbers: creating tools to improve the numbers, and constantly analysing what others have done to obtain better results. Making things measurable is an end in itself in the market economy, and is something I have learnt in the creation of the Arduino company, itself an evolution of the Arduino platform. Metrics are the thresholds in the measurements that trigger further events. A company, like a platform, is run by a simple algorithm executed on a socioeconomic assemblage with metrics we check at all times to ensure that its behaviour and performance are the expected ones. The larger the system, the more relevant it becomes to design algorithms to direct this expansion effectively.

My preception was to study assemblages from the point of view of their size, understood as expansion, and their performance, and that is what I will present here. There is no linear way to address the classification of systems, except perhaps by the total size of the assemblage. That gives us a scale that ranges from tool (the smallest), via toolbox, kit, and platform to infrastructure (the largest), as is presented—in order—in the following section. There is an opportunity to enact the power that comes with size if the number of users of an assemblage is not too great, but it might not be appropriate to invest the resources to develop a platform to support it, which in turn might also limit its ability to expand, its accessibility, or even its shared governance. As will be seen, size analysis is not the only factor to be considered when designing platforms.

²⁵ Once I settled on my thesis outline, my draft title was '1, 10, 100, 1000', in order to show how much size matters to the mindset of what one is going to design for.

Tools

According to the *Oxford English Dictionary*, a tool is ‘a thing used to help perform a job’ or ‘a device ... used to carry out a particular function’; definitions that are applicable to all of sorts of artefacts, material or immaterial, and therefore software and even processes.²⁶ The psychologist James J. Gibson (1986, 40) gives his own definition of a tool, and summarizes in a sentence how important tools are, to the point where having access to tools may have affected our evolution: ‘Tools are detached objects of a very special sort. They are graspable, portable, manipulatable, and usually rigid ... humans are probably the only animals who make tools and are surely the only animals who walk on two feet in order to keep the hands free’. Although research has shown that other species do use tools (Furlong et al. 2008), it is still correct to say that humans are far more proficient in using tools. This superiority in the use of tools happened due to our ability to acquire language (Stout & Chaminade 2009).

Pelle Ehn (1988, 392), a specialist in interaction design, in considering Ole Thyssen’s categorization of human instruments, distinguishes between ‘body, language, social institutions and tools proper’. In a sense, when I talk about tools, I am referring to what Ehn describes as ‘tools proper’—the physicalizing of work and knowledge. Tools are ‘designed, constructed, maintained, and redesigned ... design and use of tools in this sense is interrelated to the other instruments: our bodies, our languages, and the social institutions we live in’. Ehn goes further, drawing on Heidegger, Marx, and Wittgenstein to construct his theoretical apparatus. He looks at the relationships between our understanding of the world in practical terms, our relationship with labour, and the use of language games. He believes that all three approaches find the practical uses of tools to be fundamental to human practice. When we have the skill to handle it, a good tool, in Ehn’s eyes, becomes an extension of the body: ‘it is transparent to us; something that lets us have focal awareness on the task or on the material we are working with’ (1988, 393). Later in his career, Ehn moved on to investigate the process paradigm (Björgvinsson et al. 2012), to which I will return later.

26 <https://en.oxforddictionaries.com/definition/tool> [11 Feb. 2017].

When combined, tools tend to bear out the saying that the whole is greater than the sum of its parts. For example, while a hammer can be used to drive in nails and a burin can be used to engrave wood, hammer the back of a burin from the wrong side of a plank and it is possible to punch out nails without further damaging the wood. By using two tools together, we enhance them by adding new functions. Tools do not need to be physical objects. They can be found in other areas. In legal terms, for example, a rental contract is a tool used to enforce a certain relationship between the signatories. While still concrete in language terms, a contract is abstract since it does not affect the physical world through direct interaction with it. There could be a contract to dig a well, but the contract is not the one doing the actual digging. Language-wise we still refer to it as a tool that operates in law, which in the end refers to established relationships among humans, based on custom.

There are other non-physical areas where we can find tools. For example, a software tool could be defined as a piece of software used to perform a specific task in the same way we define a physical one. Examples of software tools are executable programs to operate a fast Fourier transform, software to edit video, or templates for accounting spreadsheets valid in a certain country. For Ehn (1988), computer artefacts, whether hardware plus software or only software, are tools; however, unlike the other areas, they have a very specific field of application, as they should be, according to Ehn, the ‘tools of craftsmen’ (1988, ch. 17). If computer artefacts are ‘tools of craftsmen’, do they then cease to be accessible to everyone? Were other tools, like language, initially meant to be for craftsmen only, only over time enlarging their target audience? And if so, to what extent has the general adoption of computers in every area of contemporary society changed this relationship?

Ehn’s linguistically oriented argument might seem dated. Since then, the sociology of technology has taken a ‘material turn’—a turn that accords material things the same weight as humans in assemblages (Pellizzoni 2015, ch. 72). Some of those things are computational materials that are run by software, derived from a more abstract construct—algorithms. Algorithms, as presented by Dourish (2016), are

‘materialized’ as a tool or tools in the form of one or more executable software packages (or even parts of executables). Some of the tools’ properties include the ability to alter agency through feedback loops, which allows them to respond directly to the environment. There is a clear difference between such a tool, which is context sensitive, and for example a hammer.

By talking about problematizing algorithms, Schuppli infers that we need to figure out new ways to think about the algorithmic aspect of software tools. As she says, we have ‘insufficient collective understanding as to how ... decisions [are] made’ and how they bring new power relations ‘for which we have as yet inadequate legal frameworks or modes of political resistance’ (Schuppli 2014). A new world unfolds in which control is executed through obscurity, where the lack of understanding about how technology works makes people into pawns to be controlled, turning the algorithm into a mechanism of control-by-closeness.²⁷ Are not computer tools, apart from extensions of the body, also systems of control? While Schuppli’s example is extreme in terms of the topic it deals with—the killing of humans by machines—the idea of control-by-closeness can be extrapolated to other fields. In her lecture on ‘Algorithmic Governmentality and the End(s) of Critique’, Antoniette Rouvroy (2013) explains how some of the deep learning algorithms are not only impossible to comprehend—since they build their own categories and decision mechanisms—but also impossible to criticise, since we can never know how they classify events. Besides, the algorithms will never be wrong; they will simply adjust their thresholds upon the arrival of an event. In a way, algorithms can be closed because of intellectual property limitations determined by their creators, or they could be simply impossible to comprehend, which would make them inadvertently closed.

The idea of being closed can be compared to the concept of punctualization, because of the way it is manifested. If something is closed, its inner mechanisms are invisible to us and the black box

²⁷ I am referring to the idea of ‘security through obscurity’, which is used in the computer security community to express that the best way to keep a system secure is by not publishing how it works.

paradigm applies: we know what to expect, but do not know how it will happen. In a similar way, punctualization consists in hiding a process from us due to its complexity, the frequency of its use, and so on. When something becomes ubiquitous and its complexity vanishes from sight, it goes through punctualization. This concept can be better understood by looking at the evolution of operating systems. In his essay *In the Beginning... Was the Command Line*, Neal Stephenson (1999) presents his personal journey between operating systems over the course of several years. He introduces the idea of moving from the command line interface (CLI), where all commands have to be entered by typing their names in a text box, into graphical user interfaces (GUIs) where most interactions are within a two-dimensional visual interface. Contemporary operating systems have gone through punctualization, whereby the GUI has become the predominant interface, or ‘system of metaphors’, with the different tools in the system. This hides the complexity of the artefact and makes it more accessible. While Stephenson should be considered a computer craftsman, his analysis is very pragmatic. He explains that computers are now common tools and that the operating system is also a tool, over and above which we have other tools that allow people to perform certain tasks in satisfactory ways. Stephenson’s example presents a process that could be called ‘toolification’, creating tools out of more complex systems either by zooming out (punctualization) or because of the obscurity (closeness) of the tool’s inner workings. This process invites the creation of tools for almost any purpose by mashing up different tools and punctualized systems.

The introduction of computers in the workspace as well as the home, and the different factors associated with it, including the operating system, peripherals, drivers, applications, malware, and so on, usher in a whole new understanding of the concept of the tool. Ehn (1988, ch. 16) has a whole discussion about the relationship between the tool and the user, and whether a computer should or not be considered a tool. I believe this has to do with the fact that tools—according to Marx and other philosophers mentioned in Ehn’s book—are strongly linked to labour. The actor–networks described in the various projects I was engaged in imply some sort of digital labour. For example, the SandS project was meant to involve technicians installing connected

kitchen appliances, while the PELARS project augmented teachers' work situations, and the article on the indoor location system reflects the configuration and installation of the artefact at a home (by a technician). And while indirectly addressing the topic of labour, I am more interested in the idea of the tool as one element in a larger sociomaterial ecosystem. While Ehn in the late eighties defended the idea that computer artefacts should be designed as 'skill enhancing tools for production of good use quality products and services' (1988, 407), today such an ambition needs to be differently managed. Ehn also warned against the 'tool' labelling, since as tools, 'computers alienate our lives' (1988, 408).

When Ehn wrote his book on the topic of 'work-oriented design of computer artefacts' in 1988, computers were understood as generalizable machines that could execute many different programs—simultaneously or not—to perform different tasks. Thirty years later, computation has been reduced in size and multiplied in performance (and pace) by several orders of magnitude, while becoming so inexpensive that we now have single purpose computers, even if the same machine could be used in many ways. The specificity of purpose makes the computer a tool. Hence, paradoxically, due to its abundance, the generic machine is being toolified.²⁸ Computational power is boxed into microcontrollers and microprocessors, two distinct types of integrated systems in which structural differences are blurred as technology advances.²⁹ Microcontrollers and dedicated processors allow for this new typology of single-use computers, which are different from the computer artefacts Ehn wrote about. Toolified computers are also growing in number, facilitating paradigms such as pervasive

28 At time of writing, I am also involved in the EU project 'DECODE', for which I am designing a multipurpose single board computer to be used in various single-use scenarios. To start with, I have studied the performance capabilities of six of the most used single board computers in 2017 and the first half of 2018. The study, with over 750 pages of graphs and 30 pages of analysis, indicates that single board computers are very powerful in terms of computational power, and that they can be embedded in almost anything.

29 A microprocessor, also known as a micro processing unit (MPU), consists of a central processing unit (CPU) and enough peripherals to make it work in a generic way: cache memory, input/output registers, clock, etc. A microcontroller contains a microprocessor plus program memory (typically flash), read only memory (ROM), and some specialized peripherals such as an analogue to digital converter (ADC), pulse width modulation (PWM), and so on. While it is getting harder to differentiate between microprocessors and microcontrollers, the main distinction is in their usage scenarios. Microprocessors have traditionally been used in multipurpose machines, while microcontrollers were used in single purpose machines.

computing and ubiquitous computing. The multiplicity of single-use machines requires different human–computer interfaces to the ones presented by Ehn. Additionally, our technocentric society has developed computer-based tasks or ‘crafts’ that go far beyond the labour he described. As a result, one might say that computers have gained the right to be considered tools. Or, using a concept proposed by Ehn, computers have gained a certain ‘toolness’, defined as the ultimate value of a tool. Toolness is described as some sort of contextual affordance or increased contextual capability to perform a task, because ‘tools do what we mean, not what we say’ (Ehn 1988, 403). It is in this respect important to relate this to the contemporary mindset of ‘smartness’. While toolness represents intentional control, smartness or intelligence represent automation. ‘Autonomous machines are intellectual tools, run by themselves on the basis of an internalized model of some phenomena in the world’ (Ehn 1988, 399), so that an example of autonomous machine is, for Ehn, the clock—‘Computers are autonomous machines, only much more general?’ (401).

Further problematizing the concept of smartness, Ehn critiqued artificial intelligence (AI), arguing that at the level of tool it might not be what we need to better perform a task. He sees a conflict between toolness and intelligence, between control and automation. However ‘intelligent tools may be designed to strengthen rather than weaken the user’s control’ (Ehn 1988, 404). Considering the state of the art and the development of AI as described the cognitive scientist Margaret Boden (2017), I would enlarge on Ehn’s view of computer artefacts lacking toolness, but cannot agree with his understanding of a tool as being something without intelligence of its own, where the human has to be in control. Humans may sometimes be in control of the process, or just in control of the outcome, or sometimes in control of both. This is made clear in the Schuppli’s examples (2014), Stephenson’s toolification of operating systems (1999), or Dourish’s reflections on about algorithms and AI (2016). Therefore, the concept of toolness should be upgraded if it is to describe the qualities that make something a tool in contemporary terms, including aspects such as AI and punctualization. Automation, instead of imposing a problem on control, should be seen as a way to enhance existing tools and to create new ones using embedded technology and intelligence.

In sum, tools offer intentional, direct solutions to immediate problems. Tools can be complex in nature, or the result of a fairly complex development process. Tools, redefined to be part of assemblages of humans and non-humans, could include feedback mechanisms and have an agency that allows them to perform new types of tasks that require computation. Tools' main feature remains to hide the complexity of an operation, helping us perform something effectively with as little effort as possible.

Toolboxes

'The cops confiscated all my equipment ... if you want this done tonight, I need hardware', says Elliot in the '*h1dden-pr0cess*' episode of the extremely popular television drama series *Mr. Robot* (2016). Elliot, the main character, needs a laptop, a computer scanner, a printer, some paper, a series of burner phones, and a piece of software to perform a social engineering hack, where he will obtain the location of a mobile phone he has been asked to find. His toolbox is made of hardware and software, reusable and disposable materials.

A toolbox is, by definition, a collection of tools and fungibles that allow us to perform a series of actions of a certain kind. The carpenter's toolbox will always include hammers, chisels, screwdrivers, screws and nails, a pencil, and so on, and while some of the materials are single-use, most are reusable—the tools. The carpenter's expert knowledge is what makes the combined function of the tools more than just the sum of their functions. Unlike kits, which will be described later, toolboxes do not need instruction manuals, since they are built on expert or experiential knowledge.

As Mellis et al. (2013) note, 'even users expert with a particular toolkit may remain locked in by its constraints'. This means that a toolbox (a toolkit for Mellis et al.) presents some limitations because of the affordances of the tools, and also the users' knowledge. Blikstein and Sipitakiat (2011) introduce the idea of a 'breakout model' in electronics toolboxes, which is closely related to the way the Arduino platform was conceived in the first place: users build systems using off-the-shelf components using the Arduino board as the digital controller of the various peripherals, and to permit users to do so

the boards are designed with as many exposed pins and processor peripherals as possible. However, I cannot entirely agree with Mellis et al.'s definition, as when they talk about 'toolkits' they sometimes refer to the wider utility of a toolbox, yet also use 'kit' in the more targeted sense that will be described later. Meissnet et al. (2018) do not make this distinction either, and even propose a 'meta-toolkit' as a way to produce toolkits adjusted to different contexts: 'every toolkit needs skilled "tweaking" effort to make it work in a specific setting' (10). I would argue that this mismatch between 'toolkit' and 'kit' is part of the general lack of workable definitions in the field. On reflection, I would say that the 'breakout model' gives an idea of what it takes for a tool or toolbox to grow into a platform, having as it does much of the featurelessness (Gillespie 2010) that technology needs in order to reach as many as possible.

A toolbox is made of replaceable parts; as tools and appliances wear out we add new ones. A toolbox is something you have even if you do not use it at all times. It is waiting to be used. There are professional versions, some with higher-quality tools, others intended to be used once or twice. When transposing the idea of the toolbox to the non-physical, it changes to some extent. For example, in the world of software, it is not immediately apparent how to apply the idea of disposability of materials. Software is made to be copied, backed up, transferred, and the only feature that can make software disposable is the technical obsolescence of software subsequent to the further development of the computing machines or operating systems on which the software runs. One could thus say that software, since it is attached to the operating system, which by proxy is attached to a computer architecture, has a materiality that makes it disposable. As will be seen, this is further challenged by usage needs.

There are other scenarios in which toolboxes can be defined in the same way in the world of software as in the physical world. I will present a couple of cases to problematize this, looking first at computer security, since it offers an analogy to the idea of disposable materials in toolboxes, and then at software development kits. While I have barely touched the idea of disposable software in my work, I do have experience of working with the creation of software tool-

boxes such as the original Arduino IDE or software suites for various operations in the fields of installation, simulation, and maintenance.

When looking at disposable digital material, pretty good privacy (PGP) software is a good example.³⁰ It is software that ensures secure communication between two parties. In order to make that communication secret, the parties have to exchange a series of randomly generated numbers (keys) specially created for the session. Another scenario is the keys used to identify secure service providers online. The https protocol offers enhanced communication by means of unique identifiers sitting on the servers, and those identifiers are issued by third parties to ensure no other servers on the Internet can be mistaken for the one you want to access. That unicity in the form of a unique identifier is the equivalent to the single-use nail in the carpenter's toolbox.

While PGP stands for disposability, a software development kit (SDK) equates to the reusable toolbox. While in software development SDKs are referred to as 'kits', I consider them toolboxes: a series of software tools designed to help construct software products in specific contexts. There are SDK toolboxes for all sorts of devices. One example is Vuzix glasses, an Android, powered, industry-oriented device similar to Google Glass. Even if the device is programmed using the Android programming language, which is open source, in order to compile and upload code to the device one needs a special toolchain and development environment. Another example is the Arduino IDE, the software that Arduino users need to create the programs that will be compiled and uploaded to boards. The Arduino IDE is generic in the sense that it allows for different toolchains for various types of microcontrollers to be programmed with it. In some cases, some end products can also be programmed using the Arduino IDE: the Sony SmartWatch, for example, which we managed to program by simply creating new definition files and library files for the Arduino IDE (Cuartielles & Taylor 2013a)—simple for us as expert users, of course, but not so simple if one lacks the detailed knowledge of the toolbox, which takes years to acquire. There are SDKs for all sorts of

30 PGP software is encryption technology that allows for secure point-to-point communication.

products, from programmable lamps to robotic arms. Since the idea behind an SDK is to help developers at companies willing to resell certain products in specific configurations to create their own programs to run the devices, the SDK is really a toolbox with a diversity of software, examples, and sometimes even add-ons complementing it. It is also open-ended, to be used for different purposes and in different situations.

Returning to Elliot's remark that 'if you want this done tonight, I need hardware' (*Mr. Robot* 2016), it should be noted that the series is famous for depicting very accurately the kind of tools used by the main character to hack into a wide variety of systems and obtain information. It is not the tools, the generic computer gear and office materials, but his expert knowledge in the field that allows him to make the tools perform operations that viewers might never have dreamt up. In just a few minutes he is able to make the best out of the tools to hand. It is so simple and at the same time so hard.

Kits

A kit is a set of tools and materials ready to be integrated into a preconceived design or specific context. Some kits are made of disposable parts and once used cannot never be reused, while others are intended to be reused in different combinations. A kit typically comes with an instruction manual that guides the user into achieving a certain end. The ultimate purpose of a kit depends on the context of its use. There are educational kits that are a series of experiments, while others are like flat-pack furniture, designed to give users the experience of assembling their own artefacts and the satisfaction of an almost-guaranteed success. In fact, not only analogous: one of the better examples of a kit is the flat-pack furniture box. Its small screwdrivers and screws as well as the tools included in the package are designed to last for just that one occasion. The whole flat-pack furniture industry is built on the idea that apart from the kit, everyone also has a standard toolbox with a hammer, a screwdriver, and the like. Such basic toolboxes will still need to be complemented in order to mount a shelf, put the legs on a sofa, or install a new kitchen.

Kits have a certain intentionality, as when dealing with preconceived designs. It is not that the recipients cannot change the kit's fate, but success cannot be guaranteed unless it is in the prescribed manner. Its parts and disposable materials make up an orchestrated series of sequential steps to provide the user with a somehow expected result. Thinking in terms of the sociology of innovation, as the sociologist and engineer Madeleine Akrich puts it (1992), designers, when creating kits, will force users to fall into categories or 'user types'. They also provide a vision—a clear idea of how the kit will enable a transformation. Those involved in bringing this vision to users are the innovators who will inscribe 'this vision of the world in the technical content of the new object' (Akrich 1992, 208). Akrich, who I cited earlier without discussing the intentionally educational possibilities of kits, therefore calls the final product of such process a 'script' or 'scenario': the kit is a script to guide our agency towards the vision.

When designing, I specifically avoid defining solutions using methods such as the use of personas or scenarios. This is what Latour (2006) and even Stringer (2014) refer to as 'design for representatives', since representatives never represent all of us. Designing kits with personas in mind is a form of control on several levels. First the experience is controlled, since the kit gives you the parts and tools to do one design. There are typically not two plans to build different things, but one, whether flat-pack shelves, electronics circuits, or a science experiment for children. A second aspect to this control is the pervasive dependence embedded in the kits by the manufacturer in the form of replacement parts, extensions, and complements. Finally, there is the control element of the software that runs on computational elements. For example, imagine a content management system (CMS) for a user to implement a blog.³¹ She will typically install the software from a back-office service, alter its look and feel using standard software, and upload the blog to a server. The instructions for building the website together with the actual software are a kit that will end up in a CMS that will most likely not satisfy 100 per cent of user needs, highlighting yet another element of control: software dependency.

³¹ Examples of content management systems are the software behind the blogging platform WordPress or MediaWiki (the software package used to create Wikipedia). To some extent social networks such as Twitter or Facebook can be considered CMS systems, as users can post, modify, and delete content.

The solar energy lamp discussed by Akrich (1992) was an example of a kit developed for remote places. Part of a top-down experiment to provide third-world users with lighting, the excessive concretion of the solar electricity kit made it impossible for it to succeed on this point, and it also failed to facilitate co-learning and community-building through its use (experts giving support to less experienced users). In other words, a failure in the design of these very specific sets of parts can still have consequences beyond the defined scope of the kit.

What Akrich describes is the virtuous circle of designing a good kit. A kit can be very educational or extremely practical, but if badly designed it is doomed to fail.³² Good or bad design has to be weighed against the expected outcome of the end user's experience. The kit has to offer a replicable experience, in the sense that users know what to expect, and that is why they fall for a kit rather than a more open-ended toolbox option. This is yet another view on the controlling nature of kits. While users need to be in command of the situation—to know how long it will take them to mount the shelving system or what the expected result of the educational electronic circuit will be—the kit configures not only the use but also the user. For the manufacturer of a kit, the metrics are simple: if a kit sells, it has succeeded in configuring a unified body of users. Reusability, understood as redundancy of function or materials, gives users readier control. There are lessons to be learnt from a kit that has been adopted by a significant number of users. Would greater redundancy in design, tools, materials, and documentation, which have a proven record based on metrics, serve to increase the probability of making new successful designs?

Platforms

The analyst is forced at all times to define the universe of discourse within which ‘redundancy’ or ‘meaning’ is supposed to occur. (Bateson 1972, 422)

³² It is not my aim here to discuss how to design a better kit, but rather to give a basic understanding of the various definitions.

While the toolbox requires a certain level of expertise to get the most out of it and the kit comes with the intention of empowering users and educating them, the purpose of the platform is to help the novice become an expert by participation in a community of co-learning. The platform is an assemblage in which humans and non-humans co-exist, building and controlling access to a shared dynamic knowledge base, where communication is mediated by tools, and actions are performed through kits and toolboxes (as well as generic materials). A platform is flexible by nature, especially during its formation and definition. The flexibility refers, once more, to Gillespie's notion (2010) of the 'featureless'. It is hard to say whether the platform emerges in response to a need, or if the need will be generated by the platform. At the same time, the definition of what the platform will do arises in the interaction between developers and users, and that is why its meaning is created in that exchange.

The idea of redundancy introduced by Bateson (1972, 140) is very much what defines a platform's chances of success. A platform explained through datasheets is a lot less appealing than one explained by examples of use. Datasheets or, in software, library descriptions as in the Java programming language, totally lack redundancy. They present the features of a system and list the collection of available methods and algorithms of a library, but without trying to convey a meaningful message. In software, there are tools that automatically extract the documentation from the library as it was written by the programmers. Even if this type of documentation becomes relevant over time, it is not the kind of information users feel compelled to read when about to take their first baby steps in the world of tools and processes. Yet in order to attract users to an open-ended environment, it is not enough to provide a rational purposiveness. What is required is rather a non-controlling narrative that takes into account that people, in order to experience a sense of self-control, would rather learn as they go.

It is this open-endedness in the nature of the platform, together with a feeling of territorial property, which creates a community around it. The community is the mix of the users or actors and the technology that supports them. The outcome of a community varies from case to

case—a group of people who watch and comment on YouTube videos will be different from a group who write and review articles about technology. In both cases, the creators and maintainers of the platform have to relinquish some of the control mechanisms to the users in order to transfer the degree of ownership that will retain participants, and hence make that platform into a shared environment for decision-making, relevant over time, or Björgvinsson et al.'s 'Thing' (2012). Such a continuum of user technology is an assemblage in Bennett's terms (2005), with an agency that goes beyond those that create it (Rouvroy 2013). The user thus conditions the technology by requesting different services, or contributing knowledge, time, or other resources. By becoming active members of the community, contributing to forums, helping newcomers, and so on, users gain indirect power in the maintenance of the platform. In a sense they become constitutive, and therefore part of the continuum.

Lane's reflections (2014) on the origins of the term 'democracy' (power to the common people or to the many) apply in this case too, as the platform could be defined as a democratic process, being mutually constitutive. Yet again, this involvement of people in the platform and vice versa is a self-control mechanism of a complex system. For it to continue to exist, it has to please and serve its users; it has to be designed for and by its users; it has to be commanded by its users in order not to be corrupted; and it has to sustain the becoming-user or becoming-subject in a relationally sound way, the users being 'subjected' by terms of service, state-of-the-art technology, constitutive affordances, and disturbing bugs.

Platforms allow for democratization in Saldana et al.'s terms (2016), but also in Alexander's (1977), or Feenberg's (2010). Platforms can gather and give access to data more easily and in a more pervasive way, making it available to everyone; at the same time, the process of designing the platform could be open to all actors. This concept of shared authorship is in line with a platform's ability to become a thing—here a (social) contract between parties. Their featurelessness, together with their reprogrammability, is what gives platforms the ability to reconfigure or adjust to new situations. Featurelessness also brings flexibility, understood as openness or a lack of shape,

which leaves the platform open to interpretation. From a co-learning perspective, platforms allow actors to learn through their interaction with other parts of the actor–network, whether human or not. Platforms can integrate other instances of systems such as tools, tool-boxes, or kits; alternatively, those systems can evolve into platforms when a community forms around them. Since my theme is platforms and their relationship to other entities in daily life, what matters here is whether a platform can evolve into an infrastructure, or whether infrastructures can be transformed into platforms.

Infrastructure

For Plantin et al. (2016), infrastructure studies concern the evolution of shared, widely accessible systems and services of the type offered and regulated by governments. Björgvinsson et al. (2012) bring their own definition by holding infrastructures to be intermediary objects, designed as part of a design process. Drawing on Leigh Star and Griesemer (1989), they define them as boundary objects that mediate the communication among users, professional designers, and existing devices. Such boundary objects constitute infrastructure of sorts, being material assemblages such as ‘railroad tracks, cables, or the Internet’ (Björgvinsson et al. 2012, 108) that reach beyond the event, extended temporally and spatially. Platforms and infrastructure can be compared from the perspective of scale: the latter used to be bigger as they affected whole countries, while the former, free from local ties, is on the transnational and even planetary scale. Currently, any of the platforms of large social networks—Facebook, YouTube, Twitter—has a larger technical infrastructure than a medium-sized country. Platforms also have a higher degree of emergence, for they show up at a high speed given the right sociotechnological conditions. Infrastructure, on the other hand, requires the involvement of the political class and other societal entities, and mechanisms have to be put into motion (with, say, public tenders to find technology suppliers, public studies to support decision-making, and budgeting to raise the funding needed) that are characterized by long lead times, measured in years, and sometimes full political cycles.

A concept that Björgvinsson et al. (2012) introduce is the ‘infrastructuring’ or disintermediation of a thing and its transformation into

‘commons’. My question is whether it is possible for contemporary platform-things, which typically appear on the smaller scale, to become public or shared entities. While infrastructuring in Björgvinsson et al.’s definition seems desirable, being synonymous with a co-creative ‘thinging’ that tends to boost user satisfaction when applied to platforms such as Facebook, Google, and Twitter, the speed at which those systems change is beyond the co-creative legislation of the thing itself. A slower pace in the way things adjust would only scare users away. This means that there is an imbalance between the way society creates legal frameworks and how rapidly the platforms change. This is why Mantelero (2016) and Feenberg (1991) advocate a model in which ethics and more inclusive design processes play a greater role in the design of systems. This would allow large corporations to self-regulate and anticipate potential issues.

When it comes to the economic model that sustains the most successful platforms, it goes hand in hand with the pervasively multinational or transnational nature of networked platforms. Platforms can be legally established anywhere and make money through the Internet, challenging international trade laws and the like. Much of what makes them attractive to users seems to be their mobilizing potential and the possibility to connect and exchange information with people from all over the world.

While infrastructuring talks about deformatizing or disintermediating a thing so as to leave a legacy and bring about positive change in society (Björgvinsson et al. 2012), Plantin et al. (2016) talk of ‘infrastructuralizing’ as the negative process of corporate-owned platforms taking over what should be a digital public service—the obvious example being Google’s near monopoly on searching, or Facebook’s on social media. According to Plantin et al. (2016, 3), the boundaries between ‘the two perspectives’, meaning platforms and infrastructure ‘have become increasingly blurry’, and from the perspective of Web2.0 it is hard to know what is a platform and what is infrastructure. They try to shed some light to the matter by making the distinction that while we tend to think that infrastructure is ‘essential to our daily lives’, platforms ‘are dominated by corporate entities’. One could well ask why the public sector allows

corporate infrastructuralizing to happen. Hayles (2009) mentions in her discussion of RFID technologies that the corporate surveillance we are subjected to is no longer epistemological—‘who knows what about whom’—but rather is ontological, as digital platforms are increasingly integrated with our worldview, covering the physical world and adding communication capabilities to objects which ‘are no longer passive and inert’ (48), making them equal to us humans.

People use contemporary digital platforms to communicate with one another, on a global scale and at no expense, which permits a degree of surveillance that has no precedent (Krueger 2005; Conniry 2016). The fact that platforms allow the programmability of applications on top of their data storage (Helmond 2015) ushered in a type of surveillance that is inherently part of the platform’s nature: the creation of backdoors on, for example, the encryption systems that secure the communication between users, or the geolocation information for a given user (Givens 2013). In a 2013 article, Givens introduces the kind of surveillance the National Security Agency has been exercising in recent years.³³ Since ‘megaplatforms’—as Andersson Schwarz (2017, 386) calls such transnational platforms as Facebook or YouTube—collect information from people beyond national borders, would not these platforms transform into gigantic state machines, spying on other nations? In other words, would not commercially owned platforms, through government interference, become de facto infrastructure serving the national interest? In practical terms, surveillance could escape the autoregulatory actions suggested by Mantelero (2016) or the implication of ethics (Feenberg 1991), as mentioned earlier. Is not the possibility of platform-based surveillance something that could be removed from platforms by design? Would a platform be allowed to exist if there was no way to control it?³⁴

33 While the central aspect of the article by Austen D. Givens (2013) is how new laws should not be created under emotional stress (like the ones that permitted the surveillance of platforms in the US enacted in the aftermath of the 9/11 terrorist acts), the article stresses that if technology allows surveillance, governments—or other entities—will use it as a way of gathering data.

34 Events in 2018 overtook this thesis, when it was discovered that the data of over 50 million Facebook users had been used by the company Cambridge Analytica as a way to influence the US presidential election in 2016, among others. In an interview in *Wired* in March 2018 (Thompson 2018), Mark Zuckerberg, CEO and founder of Facebook, announced his belief that companies should self-regulate when it came to privacy-related issues, but just a month later, during a deposition before

Billions of people interact using social networks such as Facebook or YouTube. The size of a machine-centric network, like the IoT, is at least one order of magnitude bigger, and thus the risks of infrastructuralizing platforms take a new significance. At time of writing, however, the challenges associated with the IoT have more to do with a need to follow an infrastructuring process of the concept than with the risks associated with it (such as surveillance). The IoT paradigm implies the advent of platforms that are perceived more as infrastructure, using Plantin's definition of infrastructure as essential to our everyday lives. Following a certain business logic, the corporations offering IoT services tend not to provide their users with interoperability, in the sense of transferability of data and applications between platforms. As long as the IoT remains a series of discrete platforms, it will be very hard for users to invest beyond the idea of participating in a network of microservices. From another perspective, Brody and Pureswaran (2014) discuss the weaknesses of the IoT as not-yet infrastructure, and in a paper on 'Device Democracy' analyse the reasons why, in their eyes, the IoT cannot succeed given the current way the Internet works. They suggest the creation of micro-exchanges to allow a platform to emerge where users could contribute by sharing use-time on their devices, but also their data through micropayments. A system such as this, the authors say, is very unlikely to come about because of the lack of technical standardization and regulation forcing all megaplatforms to share data on equal terms. Perhaps imposing a real infrastructuring, to use Björgvinsson et al.'s terminology, on the IoT's technical backbone could be a solution—not a fictitious one, as proposed by Hayles (2009), but a real institutionalization of the IoT-related Internet to force a standardization that would in turn allow for cheaper infrastructure, devices, and services as suggested by Raghavan and Pargman (2017). This would eventually have the consequence of reverse open platformization, facilitating its societal insertion, and minimizing friction for end users. Once the backbone had been standardized, it should be easy for a new type of IoT platform to emerge that could support a model like the one suggested by Brody and Pureswaran (2014). Currently, it seems the market is neither interested nor capable of initiating such a model. This is then

the US congress (Associated Press 2018), Zuckerberg suggested governmental regulation of companies dealing with user data.

the challenge, as pointed out by Hayles (2009) or Feenberg (2010): we have to publicly embrace technology in order to be able to make it a public good. Which leaves me asking the question raised by Hayles: ‘Will the current computing paradigm be co-opted as a stalking horse for predatory capitalism or can we seize the opportunity to use it for life-enhancing transformations?’ (2009, 66)

Performance and pace

What, then, of the accuracy and speed at which computers realize operations? Technically, performance and pace are determined by the number of transistors per square centimetre contained in a chip. The number of switching circuits is what defines the capability of one chip to perform more operations than another. The pace at which computing power increases is defined by Moore’s Law, formulated by Intel’s co-founder Gordon Moore (Mack 2011), which predicts that the computing power of processors will double on a yearly basis. Moore assumed that this would continue for ever.

There are two limiting factors to Moore’s law. One is purely physical. Transistors can only become so small. The other limitation, however, is user needs. Why should we continue to innovate in a field where we have all the computational power we will ever need to serve society? At some point we will decide we have an optimized processor and, as a tool, its design ought to be commoditized—in other words, it should be in the public domain for any community or company to use in optimizing its (re)production process to make it as good or as resource efficient as possible. This would be the ultimate democratization of technology, since actors would just compete on value, as functions would already be sufficient. An example close to hand is the Arduino board, where—since its design is open source—we have witnessed scores of companies competing to make cheaper boards.

Pace, meanwhile, refers to the speed at which new devices arrive on the market or become more widely accessible. Taking the development of the Arduino ecosystem as an example, when it first evolved in 2005 there were few if any competing platforms, and definitely none that included everything covered by Arduino at the time (Löwgren & Reimer 2013, ch. 6). At time of writing, the number of similar

platforms showing up on a monthly basis is overwhelming. Once people have registered the usefulness of a system, whether as a tool or a platform, the pace of development will increase, as long as there is a way for different sources to access the blueprints of such system.

The relation between sociomaterial assemblage and its computational performance points to a whole series of potential lines of analysis, especially as regards human–human assemblages.³⁵ One example is the translation of the Arduino IDE. When I started to work with education as a topic, it became clear that we would need to translate the GUI of our software to multiple languages. At some point, there was a contribution from a Japanese Arduino distributor (SwitchScience) that would permit the translation of the IDE by simply adding a standard plain text translation file. These files contain a list of strings in one language and their translations into another language—English to Spanish, for example. A program has typically one of these translation files per language. In this way, at the time of rendering the GUI, the IDE will check which strings are to be used and which translation in the IDE’s localized language, in order to send them to the GUI. For this task, thanks to the use of Arduino’s official software repository (currently a GitHub account), it was possible to translate the IDE into 40 languages in less than a week. All it took was that initial procedural work, followed by hundreds of people translating the Arduino IDE. This is an example of the performative power of a human–human assemblage, materialized in a bottom-up mobilization in which participants in a community join forces to tackle a

³⁵ Muntadas is an excellent human-only example. The catalogue of 2002 exhibition ‘On Translation’ at Barcelona’s contemporary art museum MACBA by the artist Antoni Muntadas includes short texts by various authors reflecting on the different pieces exhibited. Caterina Borelli (Muntadas et al. 2002, 252) wrote a short commentary on one of the artworks that was displayed in the show: a CNN interview with Pablo Poliaschenko, a Russian to English interpreter during the Cold War. Poliaschenko interpreted for the USSR’s president Gorbachev in his conversations with the US president Ronald Reagan that brought nuclear arms control to both countries at the end of the 1980s. In order to speed up the conversations, they decided to change the communication method from interpretation (when the interpreter listens to the whole sentence before interpreting it) to simultaneous interpretation (done in real time). This demands far more of the interpreter, who has to anticipate some of what will be said and incorporate all sort of references that go ‘far beyond the purely linguistic.’ The interpreter–president assemblage requires a certain degree of performance in order to be operative. As Borelli mentions in her analysis of the tapes of the interview, the possibility of the interpretation being objective is small, which means there is a double interpretation: first the interpreter and then the president. Such a situation gives a tremendous power to the interpreter, as can be understood by watching the interview. Reaching an optimal level of performance, both individually and as part of a group, is thus key in this kind of human–human assemblage.

challenge. In this case it was motivated by a need (having software that operated in one's native language) and not by competitiveness (making Arduino's software better than other companies' or projects' offerings).

Yet pace goes beyond computation and development. It is also related to the capability of systems to produce data. Hayles (2009), following Gershenfeld, presents an interesting definition of IoT systems as the interrelation of readers with tags (RFID) which communicate with relational databases, combined so they 'constitute a flexible, robust, and pervasive "Internet of Things" that senses the environment, creates a context for that information, communicates internally among components, draws inferences from the data, and comes to conclusions that, in scope if not complexity, far exceed what an unaided human could achieve' (49).

In looking at how two computational terms—performance and pace—apply to platforms, it is apparent that computing speed (performance) might not necessarily be a significant factor for individuals, as their user interfaces might be good enough to support their interactions with a platform, but the same cannot be said of pace, or the frequency with which new systems are developed and presented to the general public, and how frequently data can be captured from the world given a certain system. The combined performance of systems with the pace at which data is produced on platforms, including many of the IoT platforms, creates new paradigms for how to deal with information at both a technical and a conceptual level. In such cases it is correct to talk about big data, of which Hayles claims that 'the amount of information accessible ... is so huge that it may overwhelm all existing data sources and become ... essentially infinite' (2009, 47). In a way, the high pace of data generation will also require higher performance at computing centres dedicated to the analysis of the data itself. Platform, performance, and pace must go hand by hand if they are to meet user needs.

Summary

This chapter contextualizes contemporary thinking in the field of platform design. The first section explores the axes or dimensions of the discourse, including aspects of the sociology of technology, engineering, design, cybernetics, and critical theory. In tracing these axes, I noted clusters of concepts relevant to my theme: the materiality of actor–networks (sensors, communicators, actuators, modularity, affordances, embodiments, algorithms), the mechanisms of governance (distribution, democratization, accessibility, infrastructuring, ethics), the plasticity or evolution over time (always-on, featurelessness, neutrality, reprogrammability), and the design process (disintermediation, generalizability, standardization, punctualization).

In the second section on preconceptions I have concentrated on actor–networks from the perspective of size (understood as outreach), performance (their ability to compute operations), and pace (the frequency with which events and information are generated). I have explored different realizations of systems as tools, taking a new line on tools (including state-of-the-art digital technology), tool-boxes (collections of tools and materials), kits (a means of educating and empowering users), platforms (assemblages co-designed and governed by users and developers), and infrastructure (the publicly supported instance of a platform), while noting the issues involved in transforming platforms into infrastructure and vice versa.

I will next introduce the papers included in the compilation and the projects that originated them to later filter them through the above-mentioned framework in an attempt to prove the framework’s validity.



FIGURE 3
SANDS MOTHERBOARD V2, 2014

3 PAPERS

This chapter looks at the papers selected for the compilation. They fall into four groups, produced throughout the duration of the thesis. The common thread is the creation of various types of platforms, with the exception of ‘Resign Deseach’ (Cuartielles 2004), which I wrote at the start of my research as manifesto for what interaction design should be in future.

The papers describe five different platforms in which I had varying degrees of involvement. In some of them my role was merely that of interaction designer, as in the indoor location system (Casas et al. 2002, 2007), while in others I was a developer or even the head designer. My role shifted between projects, giving me the opportunity to form an understanding of how different specialists can influence the creation of platforms. Each of the papers thus presents a different researcher, developer, designer, or project, as is reflected in the written form of each. I begin with a timeline to contextualize the compilation and trace its ‘common thread’ (Chapter 1) and then introduce each paper, my role, and the outcome, concluding with the key points to emerge from the papers.

The compilation in context

The thesis compilation comprises four groups of papers from a period of some fifteen years (see Diagram 1), which report projects on the practice of research through design, the real-life deployment of wireless sensor networks, the design of connected platforms, the creation of modular electronic systems, and the implementation and testing of wearable artefacts. The common thread in all the papers is contemporary design practice, with its implementation of prototypes, testing with users, iteration of concepts, and reflection both during and at the end of the process.

(i) Research through design

D. Cuartielles, Resign desearch: The Darwinian evolution of contemporary thought species. In P. Ehn & J. Lowgren (eds.), *Design [x] research: Essays on interaction design as knowledge construction* (Malmö: Malmö University Press, 2004).

(ii) Indoor location systems

R. Casas, D. Cuartielles, Á. Marco, H. J. Gracia & J. L. Falcó. Hidden issues in deploying an indoor location system. *IEEE Pervasive Computing*, 6(2) (2007), 62–9.

(iii) European research projects

D. Cuartielles & D. Taylor. *Delivery number D2.1: Datasheets for SandS Motherboard and Modules* (Malmö: Social&Smart, 2013b) [SandS].

D. Cuartielles. *Delivery number D2.2: Report on Thin-king Appliance Manual* (Malmö: Social&Smart, 2014a). [SandS].

D. Cuartielles, E. Katterfeldt, G. Dabisias & A. Berner (2015). *Delivery number 4.2: Report on Final STEM Learning Kit with Integrated Learning Analytics for Trials* (Malmö: PELARS, 2015) [PELARS].

(iv) Exploration of haptics

D. Cuartielles, A. Göransson, T. Olsson & S. Stenslie. Mobile haptic technology development through artistic exploration. *Haptic and Audio Interaction Design* (Lund, Sweden: Springer-Verlag, 2012a), 31–40.

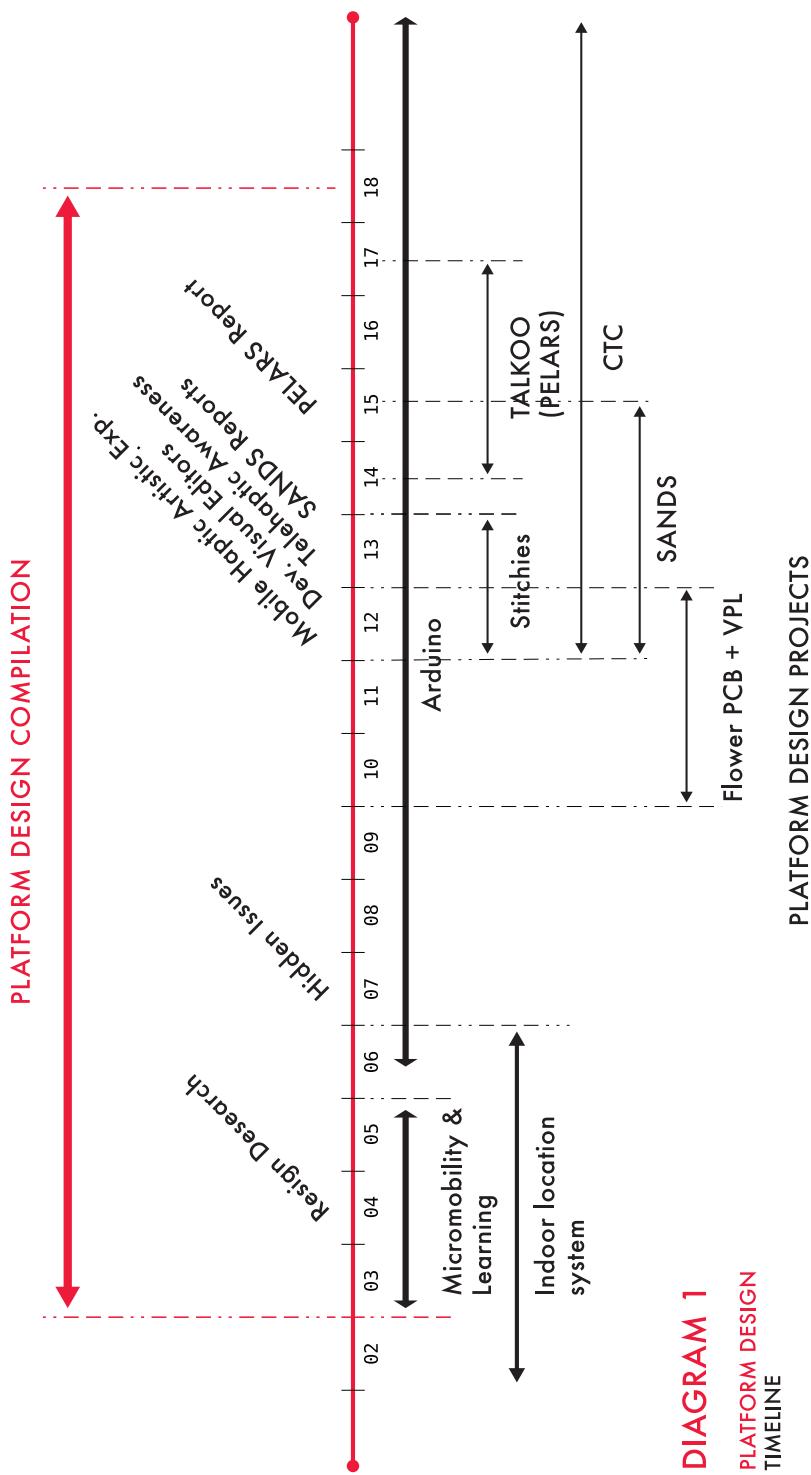
D. Cuartielles, A. Göransson, T. Olsson & S. Stenslie. Developing Visual Editors for High-Resolution Haptic Patterns. *The Seventh International Workshop on Haptic and Audio Interaction Design*, 42–4 (Lund, Sweden: HaptiMap, 2012b).

D. Cuartielles, A. Göransson, T. Olsson & S. Stenslie.
Telehaptic Awareness. In *Proceedings of the 7th conference on Tangible, embodied and embedded Interaction*, TEI 2013, 1–8 (Barcelona, Spain: ACM, 2013c).

The papers—except for the ‘Resign Desearch’ manifesto (Cuartielles 2004)—report project findings that were strongly influenced by the framework suggested in Chapter 2 concerning size, performance, and pace. For each project, I designed tools in collaboration with other researchers and users, aimed at making platforms, using state-of-the-art technology that sent data to other devices in a network. The context of all of the projects is a networked society where non-humans share agency with the other actors: hence kitchens that suggest recipes on the basis of an information exchange with other kitchens and cooks; educational platforms that invite teachers to offer help to a group of students because they detect a certain pattern of behaviour during a class; garments that connect groups of people across the whole planet; and physical interaction patterns of numerous devices programmed through visual interfaces.

Referring to Björgvinsson et al. (2012) on infrastructuring, it was the goal of all the projects to create a technology that would enhance quality of life through the intermediation of networked digital devices and services. Some of the projects had a greater platform value than others: the indoor location system was clearly intended to be an underlying technology to which services could be added; it was planned that the SandS project would set the de facto standard for how to recycle existing kitchen appliances, grouping them in the connected kitchens of the future with distributed intelligence between artefact and network; the PELARS project was intended as education infrastructure, available at all schools; and haptics experiments represent a series of kits created to enhance human communication via physical feedback, while at the same time being part of an IoT network of wearable devices.

These papers are the result of projects which, as the thesis timeline shows (Diagram 1), ran in parallel to other initiatives I also participated in. It is impossible to consider the various projects presented in



this thesis as isolated experiments, as they were done in conjunction with the testing of other concepts. Ideas, prototypes, processes, and all kinds of design tools and outcomes cross-pollinated the various projects. Therefore, in choosing the papers for the compilation, I have maximized the conceptual coverage of my work while keeping the core aspects of platform design represented in the text.

Resign desearch

I chose the first publication (Cuartielles 2004) because it deals with the etymology of the word ‘design’ and what different authors think about it. Inspired by Gaver (2012) I wrote a manifesto for what I thought I should work towards. With my technical background I had a deeper understanding of technology than other designers in my context—I could make use of technology much as craftsmen would work with other materials. In Redström’s words (2001, iii), ‘computational things can become integral parts of everyday environments’ through the use of ‘everyday materials in the design of computational things’. I tried to move beyond this view in my research, however, as I encountered, and continue to find, more and more examples of computational things becoming everyday design materials.

My essay ‘Resign Desearch’ thus set out to demystify the use of physical computing technology as a tool for creation. At the time, it was becoming increasingly common for interaction design projects to feature some specific technical element. Designers from all disciplines would learn about a certain technology and make a project from that. The claim behind this first publication was simple: given properly designed technology, it should be possible to use it the same way a tailor uses fabric or a carpenter uses wood. It should be possible to have as outcome not just one piece, but a whole collection, since it should be possible to use technology as sketching material.

I first looked at the origins of the terms ‘design’ and ‘research’ in different languages. My intention was to set the tone for what it could mean and how it could be positioned in order to occupy a unique space in the academic landscape. Design research (DR), or research through design as described by Gaver (2012), can make

use of subjectivity, and uses case studies to offer a different view or problematization of a given field. Applied qualitative research, DR's hallmark, is now more accepted as an academic discipline, because of the combination of its main ethnographic techniques with more traditional survey production (Creswell & Plano Clark 2011).

By the end of the essay I had reverse-engineered design research by using the expression 'resign desearch'. A pragmatic manifesto for a different kind of research for the design field, it allowed for playfulness, and sometimes political incorrectness. It was an attempt to name a field of design that would embrace exploration through play, surveys through intervention, and learning through observation.

The resign desearch manifesto

To contextualize the manifesto, it was written as I began my research, when I had been very active creating user experiences as part of the Aeswad art group and collaborating with the designers behind the Unsworn company in Malmö, Sweden.³⁶ The collaboration, which took place in 2003, was called 'Desearch and Revvelopment' (*sic*) and consisted of the creation of a series of outfits that were enhanced with technology.³⁷ I was the only participant in this experiment who was fully trained in electronics, which meant I was responsible for making some of the electronic prototypes we were to wear. At the same time I was working with the 'Mobility and Learning' project at Malmö University, where I interacted with students trying to figure out how different tools and spaces affected learning processes. I was then strongly under the influence of the work done at the RCA in the field

³⁶ Aeswad was the acronym for 'Am Ende Sind Wir Alle Deutsche', an art group I co-founded with Otto von Busch, Pia Skoglund, and Fredrik Svensson, all alumni of Malmö University. This group promised to itself that it would leave no written record behind, and focused on the creation of experiments and experiences to get people to discuss the things they saw. The group was moderately successful, having been invited to the 2003 Istanbul Biennale and 2004 Berlin Biennale, and had a series of repeat clients, including the Swedish government body Innovation Skåne and the Spanish company Daisalux. Unsworn was an interaction design company/studio founded by two interaction designers from Malmö, Magnus Torstensson and Erik Sandelin, which worked for museums, exhibitions, and corporate customers until it closed in early 2017.

³⁷ The work consisted of a performance in which four 'power monks' travelled the world praising the pure sources of electricity. In the performance four men dressed up as monks, their robes tied up together with electromagnets. The magnets ran off a battery pack that each monk wore on their bellies. The performers had to approach people on the street and ask for permission to recharge their batteries. The performance was repeated several times, beginning in Copenhagen at the half-machine festival in the summer of 2003. The performers were drawn from Aeswad and Unsworn.

of ‘critical design’ (Dunne 1999), which led me to write a manifesto that combined a range of ideas: the inclusion of participants in the design, the negotiation of design outcomes, and the use of unconventional methods. Given my background as an engineer, this was a way for me to trace a line of work that would see me do research through design and less using traditional engineering methods.

One passage in the manifesto reads as follows:

‘Resign Desearch’ becomes the term that refers to reaffirming the essence of searching with unconventional methods ... RD cannot afford to fail in a research process. In order to do research, we have to make design, which will probably imply the study of people. The ideology of RD is not allowing it to fail when counting with people, either users or clients, as receivers of the design result. Therefore, hypothesis testing is not a tool to be used, in order not to have an excuse to be right even if the goal is not achieved. RD has to learn both from the successes and from the failures, but then we cannot stop calling things what they are. (Cuartielles 2004, 34)

In this way, resign desearch is defined as a variety of co-design, which, because user involvement is so important, cannot fail. I specifically mentioned hypothesis testing, a research method that could come to a negative conclusion as a way of exemplifying the fact that failure with capital F is not an option if users are involved. The result might end up being entirely different from what was expected, even transforming the research question, but there is no failure as such because there was no preconceived goal.

Hypothesis testing, which may result in failure or even falsification, might provoke user rejection depending on the degree of involvement. Yves Zimmermann, one of the authors I discussed in the paper, introduced the idea of ‘success vs failure’. While ‘the designer’s duty is to produce satisfactory solutions’ (Zimmermann 1998), the term ‘satisfactory’ can be found to be controversial in the sense that, as an academic discipline, design should be allowed to adhere to the

same paradigms as the rest of academia. It should be possible, for example, to conduct design research just to falsify a theory. Design research, however, is often conducted in a different way to other disciplines, and often involves users much earlier in the research process than for example, in pharmacology, where clinical trials may take years. However, Zimmermann was not talking about academic design research; he is a practitioner and was addressing the situation when a designer working for a certain user is facing a deadline and has to provide that user with a valid solution. There are different ways to achieve this goal of finding ‘preferred’ solutions that will still satisfy the user. While participatory design, which implies far greater user involvement in the design process, was Zimmermann’s answer, another way—as I suggested in the manifesto—could be the creation of collections, providing a range of possible solutions to the one problem. My suggestion was that RD, as a sub-discipline of interaction design, was the obvious choice of field where, given the time and resources, one could always find a positive solution to a challenge. I explored this idea of collections further in the manifesto:

RD is not having single results as outcome for projects.
It will either propose to use already existing services,
devices, and tools, or it will conclude with a ‘collection’
... The day will come when instead of thinking about
prototyping as putting electronic components together,
we will take the interfaces for each cognitive nexus from
a box and will attach them together on a device ... Eve-
ryone in this business will work with collection as a basic
way of designing, and we will evaluate more complex
patterns of interaction. (Cuartielles 2004, 34)

At that point, though, I was unaware of Schon’s notion of repertoire. ‘As a practitioner experiences many variations of a small number of types of cases’, writes Schön, ‘he is able to “practice” his practice, ... develop a repertoire of expectations, images, and techniques’ (1983a, 60). Nor was I familiar with Gaver’s notion of the annotated portfolio (2012). Yet both concepts could have shaped the notion of collections, shedding light on how the notion plays out in the present. My vision in the manifesto of modular electronics is becoming a

reality, and since basic digital electronics are now a commodity, it is possible to design using standard off-the-shelf parts, rather than having to start each time from first principles. As proposed in the manifesto, this enables designers to incorporate electronics as well as users in their prototypes at a much earlier stage.

Hidden issues in deploying an indoor location system

The indoor location system (ILS) was a project conceptualized by Jorge Falco and Roberto Casas at the University of Zaragoza (Unizar). I joined as interaction designer to help with the design of the user interface that would define how people would use it, and also to define how people would interact with it as a component in an existing control dashboard of some kind. In a way, my role was to help define an application programming interface (API) for other software packages to connect to the ILS. I would then build an example, the so-called space software development kit (SSDK), a software package to augment spaces with localized sound feedback that would be featured at the exhibition at the Participatory Design Conference PDC04. The ILS would be a peripheral to what I called the space hardware development kit (SHDK), which included the ILS and computers as navigation tools.

The ILS was made of three different blocks: a series of beacons, one or more tags, and a computer. The beacons sent synchronized ultrasound pulses for the tags to receive. The tags would tell the computer using Bluetooth when they had received pulses from each beacon. The computer could calculate the time of flight for the ultrasound signals for each tag, and by calculating that block of data it could estimate the location of each tag to within five centimetres. This kind of result was far ahead of other technical solutions at the time. The software created for making all of these complex calculations, PISHA (Positioning Indoor System of High Accuracy), included an open communication port for other software packages to use the positioning information. I built a prototype to show the possibilities of such a system in the creation of soundscapes (Cuartielles & Malmborg 2004) in what I envisaged as a future platform for the artistic augmentation of spaces by sound.

The article published in the *Pervasive Computing Journal* (Casas et al. 2007) is a summary of the design work done by the whole team for the ILS. Technical features were presented along with configuration possibilities, UI design, connectivity, and thoughts about the future of a system like this one. The most important aspect of the article in the present context is the suggested reappropriation of existing spaces by new technology that was not planned to be installed there in the first instance, a topic that was among the hidden issues we discovered during the iterative process of mounting and dismounting the system.³⁸ When implementing platforms that have any kind of material manifestation, there is going to be a tension between the old materiality—the space where the platform will be deployed—and the new materiality—the combination of the space itself together with the system—which has to be factored into the design process from the start. Equally, it is not possible to anticipate all the challenges that will be faced during the design process, hence the ‘hidden issues’.

EU reports as design diaries

I supplied deliverables for two EU projects—deliverables D2.1 and D2.2 for the SandS project (Cuartielles & Taylor 2013b; Cuartielles 2014a) and deliverable D4.2 for the PELARS project (Cuartielles et al. 2015)—in the form of design diaries, including descriptions of parts, reflections on the design process, what worked, and so on. My role in both projects was as head of the Arduino group’s research, coordinating the development of technologies, handling communications with partners, participating in co-design sessions in order to conceptualize prototypes, and coordinating the academic effort behind the projects. On occasion I also acted as developer, and either designed circuit boards or wrote code for some of the prototypes. I edited the deliverables from that perspective, describing the technologies we had created ad hoc for the SandS and PELARS projects. Even if the projects related to very different contexts, the tools we created are highly relevant to the discussion here.

³⁸ When installing the PISHA system, all of the blocks that were wireless and could communicate with one another and to a computer via Bluetooth required power cables, which meant that we could have avoided using a wireless technology in its design, since we still had to wire up the whole room to power it.

SandS

The SandS project ('Social and Smart: Social housekeeping through intercommunicating appliances and shared recipes merged in a pervasive web-services infrastructure') ran from 2012 to early 2016. The main goal was the creation of an ecology of smart kitchen appliances that would share information in a social network where devices and people would interact and share recipes. There were two types: machine-centred recipes and human-centred recipes. The machine recipes were lists of sequential tasks for the machines to perform, where the commands were sent from a virtual representation of themselves down to the actual hardware. It was as if the machine's intelligence—its smartness—was placed on the cloud. The human recipes were also lists of tasks involved in certain chores or cooking. The human recipes could be translated into machine recipes, taking into account that two of the same appliances (for example, two ovens) could be entirely different in terms of heating elements or fans, but also in terms of affordances on the interface.

People could search for recipes on a social network (for example, 'remove red fruit stain') and the network would do a semantic analysis to determine whether the database contained anything remotely similar (for example, 'wash out strawberry stain') and offer the users the closest response possible. The SandS middleware—a piece of software residing on the cloud—represented each one of the devices in a kitchen as well as the sensors and actuators for each one. In that way, the middleware translated the human recipes into a generic machine recipe, and later, by adding the information specific to the device in the user's kitchen, translated it into a recipe valid for that appliance.

My role in that project was to design and manufacture a series of electronic circuit boards—a modular electronics kit—to be used to hack any sort of appliance and connect it to the SandS cloud via Wi-Fi.³⁹ My report 'D2.1 Datasheets for SandS Motherboard and Modules' (Cuartielles & Taylor 2013b) explains the nuances of

³⁹ During the development of the SandS project, I always thought of it as a platform, largely because it seemed a way to create multiple configurations of modular components. But from a use perspective, whenever a technician goes to hack another oven of the same brand, SandS is just a kit: a series of parts to configure a certain type of artefact.

this kit. It is made of a series of boards (motherboard + modules: Relay, MOSFET, Consumption, UI, Sensor); firmware for each of the modules; an operating system for the motherboard; and a piece of software to configure the addresses to each module via a personal computer. The report has sections for each of the boards as discrete objects, presenting their main features.

The work I did on this project included the design, test, validation, and manufacture of boards for all the partners to use in their experiments, which was why it had to be developed early in the process. Creating the boards presented some difficulties, since the concept behind them was unique and required a novel integration of technologies. I contributed to a microcontroller board—the motherboard—and a protocol to configure an appliance over a digital port. The motherboard was a dual-processor board with wireless connectivity that could run a state-of-the-art operating system (in this case OpenWRT),⁴⁰ but could also circumvent all the possible risks from latency in the processor when performing communication actions by adding a second microcontroller, exclusively in order to control the physical parts of the appliance.

The protocol allowed for endless configurations of the various modules, automatically assigning them addresses when connected to the motherboard through special configuration software running on a personal computer. We realized that, in order to control most of the existing appliances on the market, we just needed to create a small number of different modules in order to hack the physical actuators and sensors on an appliance and—through the motherboard—map those on their virtual representation on the cloud. The second SandS report ‘D2.2 Report on Thinking Appliance Manual’ (Cuartielles 2014a) goes beyond how each one of the boards work to explain how to set about hacking an appliance, step by step, in order to make it SandS-compatible.

40 OpenWRT is a Linux-based operating system intended for embedded devices, largely for access points to the Internet.

SandS offered an insight into what was needed in the way of hardware to create a modular electronics platform, but there was little opportunity in the project to explore the affordances of this modular system with end users. In a sense, SandS was never meant to be a project for end users; it was intended for technicians, who would hack the appliances that end users would use. Therefore, there was not much thought of how the system could be used in, say, an educational context where the same modules would be used and abused by multiple people. SandS was meant to be mounted once and left there. Therefore, being more than a platform, SandS should be defined as a kit. If there had been a continuation of the project, a deployment of a certain size, and a discussion with users about how to make sense of some of the lessons learnt while testing, this system could easily have become a platform. However, as long as it did not leave the research labs, it was just a fully functional candidate for a platform.

PELARS

The PELARS project ('Practice-based experiential learning analytics research and support'), which ran between 2013 and 2017, existed to support project-based learning scenarios where students would work in groups to create new artefacts. The various partners collaborated on the creation of a new type of classroom setting, with islands where the student groups could work. Each one of the islands included a complex technological augmentation dedicated to tracking whether the students spent time talking to one another, interacting with materials, coding, or simply staring at the screen. The islands were round tables, designed to allow the students to work standing or sitting. The desk had a computer embedded, but also a whiteboard and a simple storage for materials.

The technical augmentation of this computer-supported collaborative learning environment consisted of a computer running a software called the Collector, dedicated to the capture of data coming from the different sensed contexts. A combination of infrared and RGB cameras were capable of tracking the students' location and, using the passive markers the students wore, the direction they were looking. This allowed us to measure complex behaviour such as whether the participants in the experiments were using their hands for building

or for programming. We could also note if our subjects were looking at the screen, at one another, or at something else. The system had a ‘sentiment capturing mechanism’: a reduced keyboard made of only two large buttons that could be pressed to indicate if the students were facing an issue with the experiment or if they wanted to mark that they had a good feeling about their project. Together with my team at Arduino Verkstad AB, I created a new type of prototyping platform made of smart electronic blocks and a visual programming language that the participants would use to prototype interactive artefacts. Another of the teams created a mobile app that students used to document their progress with short texts and images. Finally, there was an analytics component to the project that could map the data coming from each sensing tool, from the electronics platform, and the sentiment buttons into an infographic that students, teachers, and researchers could use to build an understanding of the process.

Researchers ran dozens of experiments using this system at different locations, with students from different disciplines and age groups.⁴¹ They observed the processes behind the different projects and gave a qualitative grading for each one. Were the project to be extended, I would like to see the data gathered from the system used create an inference engine, an AI, that could anticipate whether students would need the teacher’s intervention if it detected the students struggling with the code.

Such a system, constantly monitoring the development of the users interacting with it and reporting to a server somewhere where the data is processed, requires a very careful approach during the design, as it can easily be classified as a surveillance system. This was a caveat we explained every time the PELARS artefacts were presented in a public forum. While the system captures information from those interacting with it, all data is anonymized by design, except for the pictures that the users will eventually capture by themselves. Simultaneously, the data was processed and rendered as visualizations designed to be conversation openers for the users and educators conducting the experiments. In the long run, what we were looking for was a way of

⁴¹ There were 52 recorded sessions with 128 users, and an additional 150 users involved in testing, making 278 participants for the academic side.

mapping the kind of processes and interactions by those interacting with the system, marrying it with a qualitative analysis of their work outcomes. The intention of this mapping was not to grade those interacting with the system, but to report in real time to the educators whether a group might need of help.

While my role in the process was not to evaluate the ethical considerations in this prototype, unavoidably there are questions about whether this kind of system should be built in the first place. It is technically possible to abuse almost any artefact, whether hardware or software, but it would require a considerable effort to abuse a system in which agency is purely informative. In the case of this design, there was no process of moral delegation as described by Adam (2005) and others—students were not to be graded because of the information provided by the system—which certainly limits the possibility of abusing the system.

Finally, since the PELARS project was not only a technical but also an academic endeavour, some of its findings were published after the project ended. Talkoo,⁴² the prototype for the physical computing experiment with students created for PELARS, featured in the *International Journal of Child–Computer Interaction* in an article on ‘Physical computing with plug-and-play toolkits’ (Katterfeldt et al. 2018) in which we summarized the experience of running dozens of experiments with the physical computing toolkit, and concluded that ‘we need to not only think about kit design but also of learning contexts in which these kits are implemented. Physical computing kits, ideally, should include appropriate guidance for their implementation in educational contexts’ (9).

My report included here, ‘D4.2 Report on Final STEM Learning Kit with Integrated Learning Analytics for Trials’ (Cuartielles et al. 2015, 17–39) covers various aspects of the Talkoo prototyping platform, primarily the modular electronics toolkit and the visual programming language (VPL). My team’s mission for the project was to create a platform that would allow participants in the PELARS

⁴² Talkoo comes from the Finnish language and refers to community collaborative work.

test campaigns to quickly prototype a simple interactive system (by quick meaning one to one and half hours). We had concluded that a VPL would permit easier, faster programming of the interaction between the various modular electronics circuits. We designed twelve of these electronic modules: LED, RGB LED, MOSFET, relay, button, potentiometer, rotary encoder, colour sensor, temperature sensor, Hall effect sensor, motor controller, and servo motor controller. The modules could be connected to one another using wires that transported not just electricity to power the boards, but also a communication bus for the boards to exchange information. Between the chain of modules and the computer was the so-called hub, a special type of module with USB connectivity that transfers the data from the USB port to the physical protocol used by modules to share data.

The communication protocol we built for Talkoo was based on the I2C protocol, with the addition of an automatic addressing system. By automatic addressing I mean the special capability of this protocol to assign addresses to the various devices interactively, while plugging in the boards to the bus—a special addition to the I2C protocol that we created for the project. The software for the project, as well as the conceptual design of the interaction between blocks and between the personal computer and the blocks, was strongly inspired by the outcomes of the SandS project.

The VPL included representations for all of the modules, but also for a series of logical modules for mathematical and logical operations on the data from the blocks representing sensors. Sensor blocks typically have outlets (a small icon at the bottom) that indicates that the block can send out data. Actuator blocks, on the other hand, have inlets (an icon that indicates that the block can receive data). Logical blocks can have both inlets and outlets. The relationships between blocks can be established by dragging (mouse click + mouse move) from the outlet of a block into the inlet of a different one. When two blocks are connected, a line joins the two blocks on the interface.

Without going into the technical details of the design, the VPL is divided into front-end and back-end and follows a model–view–controller (MVC) structure. The front-end ensures the user interaction is

smooth, while the back-end takes care of the functional representation of all the modules, their interaction at a software level with one another and the logical blocks, the control of the serial communication to and from the hub, and the sending of data to the PELARS collector for analysis. This type of structure allows for a good deal of tweaking, which this project needed because the various tests informed the design of possible improvements. We developed several interaction models for the VPL. For example, one of them made the block representing a module automatically pop up at a random location on the screen when the module connects to the hub. In this way, people did not have to drag one block into the program canvas for each module connected, only for the logical blocks. Another of the interaction models showed the blocks greyed out that students needed to perform a task, and when the student connected one of the modules corresponding to the taxonomy drawn on the screen, its washed-out colour would return to normal.

Haptics

Between 2011 and 2014 I had a formal collaboration with the Norwegian artist and researcher Stahl Stenslie, and the Swedish IxD researchers Tony Olsson and Andreas Göransson, on a project that resulted in a series of prototypes and papers on the creation of geolocalized interactive art pieces that included auditory and physical feedback (Cuartielles et al. 2012a), touch and haptic feedback-based connected garments (Cuartielles et al. 2013c), and visual programming interfaces for haptic artefacts from dozens to hundreds of feedback mechanisms (Cuartielles et al. 2012b). We published several brief notices, implemented interactive art pieces, and experimented in the creation of various electronic boards to augment digital devices such as mobile phones with haptic feedback. We made a series of different designs tracing a clear evolutionary path in the way we thought about designing embodied haptic interactions.

The collaborative process that result in the publication of ‘Mobile haptic technology development through artistic exploration’ (Cuartielles et al. 2012a) started back in 2008, when Stenslie showed the rest of us an early prototype of a sound- and haptic-augmented

wearable piece that required a laptop to perform. We continued by improving on that early design by adding a board we called the ‘Dusk’.⁴³ This first technical collage was used by Stenslie in two wearable artefacts: *World Ripple*, which gave the visitor a geolocalized experience, walking through a city searching for the places where a number of haptic experiences had been placed; and *BlindTheatre Bodysuit*, worn indoors as part of a performance at the Norwegian National Theatre in Oslo.

Göransson and I had been writing a library for Android phones to communicate with the Arduino Bluetooth board, SweetBlue (Cuartielles et al. 2011), and a book that explained how to connect Android devices such as phones and tablets to the Arduino Mega ADK board (Göransson & Cuartielles 2012).⁴⁴ We thus had considerable experience of the creation of interactive pieces that combined Arduino boards for the physical interaction with Android devices to run the intelligence, as well as connectivity aspects of prototypes.⁴⁵ We suggested that a simple improvement to Stenslie’s prototype was to use a mobile phone to create and control the experience instead of a full computer, and an Arduino board to control the haptics he planned to embed in various garments. This became *Psychoplastic Bodysuit*, first exhibited in Ljubljana in 2010. We evolved different designs: the Flower, a device that could run 80 motors from a single processor; the Stitchie, used to create a simple collar so multiple people could communicate with one another, a design that we termed ‘telehaptic awareness’ (Cuartielles et al. 2013c); and a full-body suit with 120 touch points (Stenslie et al. 2014). My contribution in the projects was the design of the electronics and firmware for one of the prototypes and the creation of the fundamental concept of a communication protocol for some of the later designs.

⁴³ The Dusk board had been created by David Sjunnesson, a former student of mine who had collaborated with Göransson, Olsson, and myself on a project ‘1scale1’, dedicated to the production of prototypes of products and installations.

⁴⁴ A now obsolete design for a board that could communicate using Bluetooth, for example with Android phones running Android Froyo and later.

⁴⁵ The Arduino Mega ADK board was born from the Google design, the Android ADK board. It was intended for experiments with the connection of devices—via a USB cable—to Android phones and tablets in order to trigger the applications that could use the data generated by the external devices or accessories.

Psychoplastic Bodysuit, included in the ‘Mobile haptic technology development through artistic exploration’ paper (Cuartielles et al. 2012a), required us to design a new flower-shaped circuit. It was worn on the back, largely because that was where the laptop computer had been in Stenslie’s earlier design. Resembling a flower, it consisted of three different boards: the ‘Cuore’, a remake of the Arduino Bluetooth, but hexagonal in shape; the ‘Leaf’, a board that could host up to 16 vibrating motors; and the power leaf, which carried the power supply needed to provide electricity to the whole system. The way *Psychoplastic Bodysuit* worked was ‘much like *World Ripple* [as it] used GPS to geotag haptic data into zones defined by spatial coordinates. When entering predefined zones, users would trigger and sense the space through different combinations of vibrotactile patterns and sound’ (36). The Flower was our first attempt at a platform for the creation of auditory–haptic feedback, where creators could combine geolocation as input and two different kinds of outputs. We did not produce many prototypes, but the intention of making something transferable was there. It was in effect a continuation of the work I had been doing in 2003 on the ILS (Casas et al. 2004, 2007) and the SSDK (Cuartielles & Malmborg 2004).

At Google IO 2011 (Google Developers 2011) it was announced that it was now possible to develop physical accessories for the Android operating system for phones and tablets by means of a new open protocol, the Android Open Accessory (AOA) protocol (Göransson & Cuartielles 2012).⁴⁶ The novelty was that it would be possible to send and receive data from accessories using a USB cable while the device was charging, and that when connected the accessory would share a URL that the Android device could use to download an application to allow it to start communicating with the accessory. To encourage the Google developers’ community to make prototypes using this new protocol, Google gave away a thousand Google ADK boards. These boards were a derivative design of the Arduino Mega, with the addition of a chip (with the function of an USB host) that permitted communication with mobile phones via USB. This gave us

⁴⁶ Google IO is the Google annual developers conference, where the new features of the Google ecosystem of platforms and tools are announced.

the idea of making a reduced version of *Psychoplastic Bodysuit* that would not need Bluetooth or as many vibration points. The aim was to make more prototypes so that more people could experience it.

Sense Memory experiment [was a] simplified the system in order to experiment with an outdoor theatre involving multiple users ... the user experienced invisible ‘sculptures’ by walking around in selected areas. New sets of haptic sculptures/expressions were geotagged onto 30+ zones placed around the square. Once the user enters one of the invisible geographic zones the sculptures came alive inside the cape as a combination of binaural sounds and vibrotactile patterns. (36)

There came a point in the construction of these experiments that we realized we needed generalizable prototypes—easily replicated and scalable so more users could test them. This was our first attempt at a platform done in a conscious way.

Another outcome relevant here was the creation of a series of visual editors for haptic experiences.⁴⁷ I had a hand in their conceptualization, but all technical credit goes to Andreas Göransson, who did the coding. The evolution of the various editors was documented in ‘Developing Visual Editors for High-Resolution Haptic Patterns’ (Cuartielles et al. 2012b), where one of the main contributions to the idea of multiple-output feedback design was the invention of a timeline-based system to define the feedback from a multiplicity of haptic points. The fact that we had such a system available was crucial to its identity as a platform in the making. By making the visual haptic editor possible, we enabled creators to easily build their own haptic patterns, and in later versions of the software to add them to specific locations on a map.

The experience of creating the initial haptic prototypes, together with the concepts emerging from the different iterations of the software

⁴⁷ In 2005, while a guest researcher at the Interaction Design Institute Ivrea, I worked with my brother, Diego Cuartielles, on the creation of a platform to design feedback mechanisms using motors. Unfortunately this work was not documented publicly.

created to program the behaviours of the interactive garments, led us to create a new type of electronic module, which we called the Stitchie. Analysed in a couple of papers, the best example was presented in the short paper on ‘Telehaptic Awareness’ (Cuartielles et al. 2013c), presenting a system that used four Stitchies connected to a Bluetooth module to create a connected garment. By touching the boards inside the garment, the wearer sent a signal to all of the other connected garments and provided force-feedback to the other wearers in the area.

The Stitchie was a new design of a single touch-sensitive actuation point that came from thinking about how to create systems that would allow for easily expandable, configurable haptic systems. The basic design consisted of a simple board to simultaneously sense whether someone was touching it and vibrate in case it received a command through a communication port. When touched, the board sent out an indication through the same port. It was the resultant circuit, in an electronic design by Tony Olsson, which was the Stitchie. It could be connected in series in an almost endless chain, only limited by the amount of power needed to run a number of motors simultaneously. I contributed the protocol that made it possible to connect the Stitchies to one another, while Göransson made the phone application that allowed the systems to share information about the various touch points in interaction with a standard message queue telemetry transport (MQTT) server. The outcome of the design was two different experiments: a necklace with four touch points that we wore for an extended period to establish how it felt to be physically connected to a group of people over time (Cuartielles et al. 2013c), and a full body suit with 120 touch points (Stenslie et al. 2014).

In a sense, the Stitchie board, together with a Bluetooth chip, the software, the phone app, and the MQTT server to exchange data from different systems were together a platform prototype. It was open to endless configurations, but we only had so much time to try things out.

Summary

In contextualizing the papers included in the compilation, in this chapter I have explained the core concepts and relevance of each in turn, classifying the prototypes and projects they analyse according to the parameters described in Chapter 2—size, performance, and pace. While a useful way of classifying the outcome of a design process, this kind of framework has its limitations because it passes over other relevant aspects of the projects. For example, what happened with the technology chosen for the SandS prototypes? Should technical choices be based solely on the expected growth of the system? Can a size-centric analysis be sufficient to inform my design process, and especially whether a certain technology is *good enough* to implement a platform or not? How do other factors such as the cost of systems affect the technology’s dissemination? Are democratic issues, understood as access to the technology and governance of systems, addressed at all using such a toolset? And what about the system’s sustainability? Is there a way to ensure it works for as long as users might need it given the current framework?

An analysis of the projects is alone insufficient to identify whether any of the projects could have become a platform. The papers understandably do not capture any of the nuances of the work on other design process projects running in parallel, which were not unimportant for the development of my understanding of platform design as a field. The timeline (Diagram 1) contextualizes not only the sequence of publications, but also how they related to other projects. The issue is that the papers do not mention those other projects, despite being reliant on them (for example, Arduino was needed in the creation of the SandS prototypes). Therefore, I will return later to the projects and design processes that are relevant here, but that have not yet been published (see Chapter 5).

The fact that relevant experience is lost when not documented in an academic format is reason to analyse what constitutes a ‘body of work’ in a field such as interaction design, where physical prototypes, devices, ethnographic work, manuals, code, design diaries, and so on are decisive for the production of knowledge. The effort invested by

a design researcher in a prototype such as Talkoo for the PELARS project does not lend itself to publication, and the same is true of the kinds of interactions it makes possible, even though trialling it gives a better understanding of what the prototype does. While a prototype is not a valid academic outcome if it goes undocumented in the form of a publication, the special challenges of practice-based, design-oriented, experimental research and development must still be borne in mind, given the conditioning of the academic format, and it is worth reflecting on the need to integrate theory and praxis more closely. Chapter 4 presents the methodology that can contribute to this, evolving as it does in the context of participatory activist research projects, where the difficulty of documenting what happens in an actor–network is a pertinent issue.

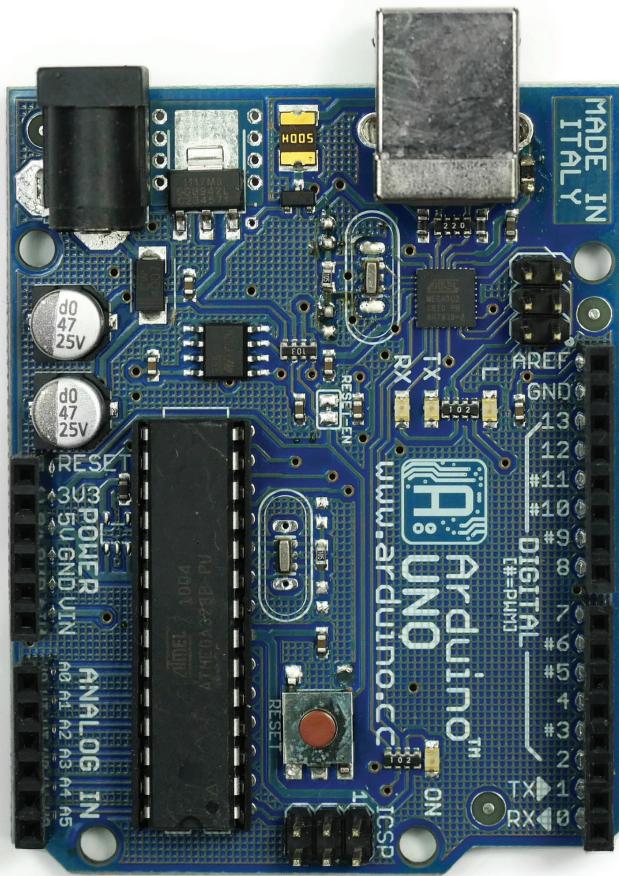


FIGURE 4
ARDUINO UNO BOARD (PROTOTYPE), 2010

4 METHODOLOGY

In this chapter I reflect upon the methodological tools I have used when developing the projects analysed in this thesis. A number of tools were chosen once already engaged in the design process. This late selection—and sometimes design—of the tools was dictated by the size of the assemblage I was dealing with, as I was just one of many actors in a large network, to borrow some actor–network terminology, and it was hard to anticipate potential issues such as the power dynamics among participants, the availability of technical resources, and so on. Over the years, I have developed a methodological toolbox based on four main clusters of methods: reflection in action; openness; use, participation, action, and activism; and inclusive or multiple prototypes.

These ways of approaching the projects all stem from my choice to apply a research through design (RtD) methodology (Gaver 2012) in my work. I have already noted the existence of RtD when talking about annotated portfolios and repertoires, a notion taken from Schön (1983b, 60) (see Chapter 3). While it is possible to find earlier references to RtD, Gaver’s is the most succinct description in his 2012 paper ‘What should we expect from Research through Design?’

One key aspect of large-scale prototypes of the type discussed in this thesis—the Arduino case being the most significant one—is the user–researcher relationship. The prolonged interaction with users of different sorts—project users and end users—addressed in this chapter has helped me develop a specific understanding of possible roles and the kind of knowledge an expert in the field requires when performing studies on existing platforms or large prototypes of upcoming platforms.

Reflection in action

Platforms, as structures implemented in and on top of actor–networks to facilitate relationships between actors, are in constant change. Platforms do not have an end goal—they are non-teleological—as they are in a process of continuous formation through use. Thus, the only way to research platforms is by using them, by becoming a user; researching a living platform is only possible through direct intervention with what is an ever-changing process. Reflection in action (Schön 1983b) is a working method that allows for the building of an understanding of the research object while interacting with and through it. Schön describes the role of the professional when dealing with a complex situation, which could well apply to a platform:

the practitioner ... responds to the complexity ... in what seems like a simple, spontaneous way. His artistry is evident in his selective management of large amounts of information, his ability to spin out long lines of invention and inference, and his capacity to hold several ways of looking at things at once without disrupting the flow of inquiry. (Schön 1983b, 130).

According to Schön, the feedback in an analysis like this occurs simultaneously with the praxis, through language and in a dialogic process.

As researchers we need a way to trace the relationships between the actors in an actor–network. For example, if some users participate in the creation of a new software feature and they would like others to know how to use it, they need a way to inform others. Documentation (for an understanding of the function), feedback (for collaboration on the co-design), and reflection (for the discourse about the platform) are the mechanisms of a platform, and relevant to our research process because they register the traces of the relationships between the actors in the network, and not just the meta-data (who talked to whom and when), but also the intentions and results of the different processes (the actual information exchange between parties).

Looking at the way I selected the various cases that would make up my research portfolio, it is plain that mine was a hands-on approach to all the projects, including those referred to here but not in the compilation, such as the building of the Arduino community. My research has spanned code and electronics for building interactive devices, art installations, online tutorials, university courses, academic year long education programmes for schools, embedded tools, and platforms. My fieldwork, documentation, feedback, and reflection has taken various forms, depending on the context. In the creation of the Arduino project, documentation was very much a joint effort—developers and users documented uses, cases, and processes at all levels—and leaving aside personal communication by email, mostly standardized, using online publication tools. The European research projects (SandS and PELARS) had a twofold documentation mechanism: internal, where data was gathered and shared with the other researchers; and external, where information was published once it had been filtered, processed, and organized. The documentation for the indoor location system was generated by each participant individually, and notes were shared when necessary. Finally, the haptics prototypes were documented with academic conferences or art venues in mind, using standardized online tools.

Erkki Huhtamo, a media archaeologist, has coined the term *thinkering* (a portmanteau of tinkering and thinking) for the ability to construct knowledge of materials in flux (Huhtamo 2010). Schön's reflective practitioner is Huhtamo's thinkerer, yet while the former looks at practitioner designers (Schön 1983a, 1983b, 2001), the latter looks at artists. Regardless of the difference, both arrive at the same conclusion: there are ways to know through praxis. Thinkering is a default modality when developing projects. While preparing the technology, one looks for features and tries to anticipate as much as possible, but then things may happen on site compromise elements of the original design. At the same time, however, the designer gains experiential knowledge that cannot be acquired any other way.

This raises a question of epistemology. What constitutes acceptable knowledge? Is thinkering a field that could be contextualized as a positivist research paradigm? Or should it be consider more

as coming from the constructivist school? According to Osterman (1998) ‘reflective practice is a professional development strategy with roots in the constructivist paradigm’. Therefore, the reflective practice, as well as thinkering, should be understood as adhering to an inductive paradigm. Descriptive research can be either quantitative or qualitative. It involves collecting data from events, which are then tabulated and probably displayed in graphic format to aid understanding of the data distribution. Interventionist research, which is much closer to thinkering, ventures into the realm of the applied, and requires the use of qualitative research strategies with a special sensitivity to the constructedness of data.

Reflection in action is an attempt to bridge what happens on site, while working with the case, and what happens afterwards, when reflecting on the work. Research through design (RtD), which builds on the cases collected throughout a research process (Gaver 2012), is thus very similar to reflection in action. Gaver suggests a threefold strategy when forming an understanding of design as a tool for inquiry and knowledge production. After reviewing some aspects of the philosophy of science, he concludes that RtD does not follow an existing standard procedure with which to build cumulative knowledge; indeed, he invites the reader to consider the design process as having a generative behaviour. His third proposition is that practice-based research should look at theory as the ‘annotation of realised examples’, which is where the annotated portfolio comes in. Gaver suggests that the design research community should avoid standardization and ‘take pride’ in the speculative nature of the field, and its ability to produce results in the form of new artefacts. To the suggestion that case-based research be used to create theories, Gaver argues that theory should ‘annotate’ the examples, but not ‘replace them’.

RtD should be valued because of its ‘ability to continually and creatively challenge status quo thinking’ (Gaver 2012, 938). Therefore, forcing it to fall into structured means of classification, or standardized ways to measure, is, as Gaver (2012) would have it, not the best way to optimize the outcome of research through design. This continual, creative challenging is very much in line with Latour’s idea

of description (1996). The role of the researcher is not necessarily to explain, but rather to trace actors under certain circumstances when performing in and as networks.⁴⁸ There is another strong link between Gaver's RtD and Latour's material and circumstantial tracing. Gaver mentions 'wicked problems' (a concept taken from Rittel and Webber) as circumstances that cannot be theorized in advance when working with design (Rittel & Webber 1973). It is therefore impossible to work with a Popperian hypothetical-deductive model, based on a systemic trial-and-error falsification process. Instead, designers create solutions as they go, and will eventually create new theories as well, the ultimate goal of which cannot be the falsification of a theoretical hypothesis that did not exist in the first place. Gaver summarizes this by saying that 'the synthetic nature of design is incompatible with the controlled experiments useful for theory testing' (2012, 940), once more confirming the inductiveness of RtD.

RtD accords a major role to the artefacts that result from a design activity. Annotated portfolios are a way to bring together the 'ultimate particulars of the artefacts rather than abstracting across instances as language does' while still allowing for 'extensibility and verifiability' (Gaver 2012, 937). On the other hand, no matter how valuable these portfolios may be for the creation of generalized theories, their role should be limited to inspiration and annotation. The artefacts are boundary objects that can be fed back into the process.

The outcomes of the creative endeavours I have been involved with, along with their side projects, have given me a rich collection of cases—in number and content—open to theoretical reflection. However, I face the issue that in some of the processes in which

48 Comparing actor-network theory with Euclidean mathematics, one can say that a network is a closed, three-dimensional form that has a changing structure over time, where other dimensions are represented by changing the colours of the different parts of the volume of the shape, for example. The researcher traces the network at a certain time and under specific circumstances. The trace is a written description of the case, what in maths would be bi-dimensional planar graph—a projection of those points of the shape that the researcher has been in touch with. Each point would be an actor, and the lines joining actors together would be the relationships between them. Extending the analogy to Gaver, his is an algebraic construction of design. An artefact is a point on the place; a portfolio (whether from an individual or a group) defines an area. Comparing different designs within the set creates a design domain, defines its boundaries, and helps frame the designer's opinion on the topic. Making design collections helps configure the understanding of the field by the variability and invariances of the artefacts.

I was involved, the speed at which the project developed and the amount of data generated made it impossible for me to document it fully. Even if I did not have a standardized way of collecting information, though, I still gathered experiences in different ways. For example, in the Arduino project, most information is publicly available in the Arduino forum I put in place, while in the SandS project the documentation took the form of a research report and an instruction manual, so that other researchers could repeat the same kind of experiments. Documentation was not my role in the haptics projects, but my colleagues did a good job of registering the results of the work. What matters here is not the collection of data per se, but the fact that each one of the cases referred to, by being included in the compilation, is now part of my annotated portfolio. They are the cases that comprise my expertise, in this case in platform design.

RtD assists in the creation of theories in five distinct ways, again according to Gaver (2012). The first is the clustering of conceptual statements from the analysis of multiple cases. Design researchers draw on concepts and models from other disciplines to inspire new designs and articulate existing ones. If the notions brought into the design field are adapted to the new context, the process becomes ‘translation’ and can generate new concepts. A third way of producing theories is through the creation of manifestos. These look at contemporary practice and how to alter it to generate a desirable future. Manifestos ‘borrow theories to justify’ themselves. Frameworks for design are described as manifestos, but are involve less theorizing and are less normative; rather, the framework implies an ontology of factors that leads to a theoretical result. Finally, there is the preformatted process of standardizing practice, evaluation, and outcome. This last method, established in the discipline of human-computer interaction is not to Gaver’s liking, as we have seen.

I wrote a manifesto as a prelude to my research studies (Cuartielles 2004), while I produced a series of cases where I designed or co-designed specific tools and platforms to be used by different types of users in a variety of contexts. Even if I did not initially consider producing an annotated portfolio, I noticed how my work gradually adopted that form. Portfolios range from design reports to user

manuals. Documenting a platform like the one in the SandS project (Cuartielles & Taylor 2013b), which is made of six different boards and complex software on different machines in the users' homes and on the cloud, in just a couple of pictures and a short text would have been a very challenging exercise in synthesis that would have omitted many of the nuances of the design. By developing a process of annotation (explaining, in retrospect the relevance of the design as contextualized in my work process), and by tracing the common thread between the projects in the present thesis, I have constructed theories and research openings to build upon. In RtD, theory is a boundary object between cases, and not just the abstracted superstructure above them.

Openness

My understanding of openness as a method stems from the very pragmatic approach to design where there is no interest in making something that is not meaningful or useful. This is the line taken by Björgvinsson et al. (2012) in their concept of the 'Thing', where design exists to be used as much as possible, whether in a second life (upcycling, as in the case of the SandS project in which we upcycled home appliances), in a modular approach that allows the integration of one's designs into other people's (as in the creation of Arduino, a platform of modular electronics), or by whatever means will ensure a more sustainable design (see Chapter 5). From this perspective, an open design implies an element of democratization, as in Alexander et al. (1977), with the openness guaranteed by a distributed process, sharing responsibility for the design with the community as a whole.

A community collaboration on co-creating a platform comes with a caveat: ownership of the system so created falls to the network itself. In her essay 'While Waiting for the Third Industrial Revolution: Attempts at Commoning Production', Seravalli (2014) studies the example of Fabriken, a makerspace in a cultural centre in Malmö, Sweden, as a complex assemblage of humans and non-humans, and how sharing of spaces, tools, and knowledge works over time, and concludes that it is an example of 'commoning'—a process of con-

tinuously re-evaluating the notion of ownership, access, and the use of resources. In a space like that, where it is impossible to work in isolation, the distribution of authorship becomes a useful method of generating relationships between actors.

Published in the same anthology as Seravalli, my essay ‘How Deep is Your Love’ introduced the idea of ‘radical openness’:

Arduino’s designs are not radically open from a process viewpoint. When we made Arduino, we built upon years of accumulated experience in teaching electronics to beginners. We did not sit down with users to create our first boards. We knew what needed to be done. The participatory design aspects of the creation had been done in the form of fieldwork by all of us. We decided back then that using expert knowledge about users, about technology, and about the relationship between users and technology should become our development method.
(Cuartielles 2014b, 163)

I would distinguish between ‘radical openness’ and plain ‘openness’. The difference between my radical openness and Seravalli’s communing lies in the concept of authorship. Once Arduino went from being an idea to becoming a product with intellectual property owned by a company, there was a differentiation between ‘us’ and ‘them’. Us, the designers, had to consciously distribute and share knowledge with them, the users or community. The first Arduino boards were born from expert knowledge, published as a service to the community, but there was no radically open process where users were involved, influencing the shape of the board or the basic firmware. Those were decisions made in isolation, and only later shared openly with the rest of the community.

When co-creating a piece of software, being radically open in a network that operates over a large geographical area—potentially even globally—is fairly simple. If all participants have the same compiler and the same source code, all of them should get the same

binary file. In the world of hardware, this kind of co-creation is less easy, because it requires a great deal more information. Thus there is another take on radical openness that has to do with the openness of the production process: it is not only the human process that has to be open, but the non-human element in the assemblage should also be traced in a way that will make it replicable. This is something that the open-source hardware community has discussed extensively, especially how to manufacture a certain design. Making something radically open is more than getting access to blueprints. There are lists of materials, manufacturing processes, sometimes even the thickness of the copper on a printed circuit board: all of it matters for the proper functioning of the artefact, and just seeing at the logical design is not enough. This concept has been incorporated into the open hardware licence models (by extending the licence coverage to more than just the logic design) as well as in the community list of best practices (Ayass & Serrano 2012; Open Source Hardware Association 2012).

As in SandS, PELARS, the collection of haptic feedback prototypes, and all the other the projects I have been involved in, with the exception of the indoor location system (Casas, Cuartielles, Marco, Gracia & Falcó 2007), my involvement with users—my design practice—was predicated on the principle of radical openness, as the designs were custom-fit for the project and required the direct involvement of everyone participating in the projects in order to be really effective and deliver the projects on time. It is important to mention that, as a pragmatic researcher, it was relevant for me to take a practical approach and gain a good understanding of the existing socioeconomic dynamics. Openness became a tool to exchange insights into the process with others, a knowledge-trading mechanism, the access card to a reading club—except that the books, as in every sociotechnical assemblage, were written on the fly.

The results for all of the projects in this compilation (besides the location system) were generated under the same premise: the creation process was open, and whatever resulted from the projects, both tools and platforms, had to be released as open-source software and hardware. There is a political aspect to this which I omitted from my

manifesto (Cuartielles 2004) due to a lack of experience. I have now explored it in retrospect by looking at the implications of openness—availability of sources, transparent processes, and distributed collaboration—all of which should be expressed in the shared definition of community practices. Seravalli's commoning (2014) is thus very close to my radical openness, for it is not just a question of someone making a design and licensing it openly for others to use, but rather of opening the conversation to the rest of the actor–network and sharing more than just the generated knowledge. Radical openness has this other layer of meaning, an openness about processes and intentions, because it aims to involve others in influencing the design process. Transparency is all.

But how can openness as a method be operationalized in contemporary society? Which are the boundary objects that will enable radical openness to be deployed on systems? Are good intentions towards your fellow beings in the actor–network sufficient? My experience, in retrospect, is that this is not always the case, but there are mechanisms or boundary objects that make it possible to repurpose openness as an inventive methods of sorts (Lury & Wakeford 2012), where the licence for the design becomes the enabling device that catalyses the process. It is to open licensing as a form of methodological activism that I will now turn.

Licences are contracts

The question of licensing is key to contemporary design practices, especially if digital components are involved in the process, whether hardware, software, or documentation. Licences are devices that can help open up a design process, as we have seen, and they set the rules of sharing in ways that have become standard in contemporary society. Tools such as creative commons (Lessig 2004, 282–6), free software (Stallman 2002, 92–4), or open hardware licences (Ackermann 2009; Ayass & Serrano 2012) offer designers, engineers, artists, and even whole communities of co-creation and co-learning the legal tools—the so-called ‘open licences’—to share their creations, whether made individually or collectively. Open licences are relevant because people (users and other creators) want to know about the possibility of reusing others’ designs (Candeira 2006). For example,

Arduino's support system registers weekly emails where people ask about the possibility of, say, making a book using Arduino boards, or whether they would have to open source their own Arduino-powered products because they are using an open-source development tool.

Open licences work in the digital world, but also in the material one, especially if there is the possibility to express the physicality of something as a digital file. For example, the artwork for a printed circuit board can be stored in a human-readable digital file, and it could eventually be protected under the laws of copyright (since it is human-readable), which is what gave the Arduino founders the idea of licensing Arduino board's blueprints under a creative commons licence. This is what led Arduino and many of the Arduino derivatives, such as the Arduino Lilypad, to use such a licensing schema, as Powell (2012) noted when analysing the history of open design.⁴⁹

But why open licence as a method? Are not licences mere tools? What is that they are proposing that is so different from the existing intellectual property tools? Open licences barely existed in the twentieth century, and by the time the licences appeared, society had been operating for a long time with a legal framework that seemed to be functional. If you invented something, you could patent it and then decide if you wanted to allow others to exploit the outcome of the design; if you wrote a book or a song, copyright law automatically protected you. While an actor–network can take out patents as boundary objects to control the relationships between creators, manufacturers, and users of technologies, these legal mechanisms hinder the possibility of having more open processes in place later. As an activist researcher, could I reasonably expect people to participate in a co-design process on condition that I would take the results, patent them, and retain the right to exploit the results?

In the realm of immaterial products and services, open licensing has been strongly optimized. From an IxD perspective, processes have been relieved of any kind of friction that might stop people from using them. The creation of open licences has been automated behind APIs

⁴⁹ Open Source Hardware Association 2012a has an excellent summary of its history.

in order to minimize the burden for creators, but also to enable digital platforms to take advantage of those creations (Creative Commons 2017). For the platforms' part, the use of APIs allows for the fast population of content-centric platforms, such as YouTube or Flickr, with perfectly reusable, broadcastable content. Flickr (Yahoo Help 2017) or YouTube (Brown 2017), offer the possibility of licensing content in different ways. Price—open, free licences cost nothing, unless you want to enforce them in court—as well as the possibility of doing it oneself or in an automated fashion may also be reasons to use open licences.

In design and engineering circles there is a certain aversion to patents, since they have proven unworkable for smaller players in competition against large commercial interests (Ackermann 2009, 194–195), which in some cases own thousands of patents (Hixon 2013). If intellectual property protection through copyright and patents makes no sense for creators, it makes even less for receivers of the designs (Hippel 2005, 114). Why then bother using patents to restrict the use of a tool or system to those we choose? Besides the more practical aspects of non-open licensing schemas (price, time to market), there is also a fundamental moral aspect that speaks against them. Imagine you co-designed a device, process, or service within a community as an open process. While it would be possible to patent something as a group, how would that form of intellectual property protection play out in the context of infrastructuring and the creation of things? I believe that openness as a method demands that one anticipate the questions that participants might have in this regard, and therefore familiarity with the shared ownership models for intellectual property will be key in creating the right atmosphere of cooperation between participants. Therefore, licensing is relevant to openness as a method, and should be introduced to community participants early on.

Use, participation, action, activism

When users and designers engage in the co-creation of platforms, they take on different roles depending on the size of the actor–network, among other factors. These roles are not a constant, but like the projects themselves they evolve. The one that seems to be fixed,

though, is that design researchers assume the intentionality of being willing to pursue a certain project, which turns them into catalysts in the actor–network. As the project evolves, users too turn into active agents. Participation in a network occurs at the intersection of two different axes: the user axis and the designer axis. Users in ideal communities are expected to be active participants. As Alexander et al. (1977) remark, the distribution of power is possible if working with communities small enough for all members to be engaged in the governance of the system. As platforms mean many actors can interact at once and participate in the process of continuous construction of the platform, it also becomes easier to be a mere user whose voice is harder to hear in the crowd. In a different strand of analysis, I look at the differences between action research and activist research as co-existing methodologies for designers to enrol in co-design processes within platforms or as a way to develop and maintain platforms. I also revisit the role of the expert to gauge how designers might best manage their own expertise in processes such as the ones analysed in this thesis.

First, the existing design methods for participation. The computer scientist Gerhard Fischer (2003) compares user-centred design (UCD) with participatory design (PD) as a way to introduce meta-design and its defining activity ‘underdesign’ as a way to create design spaces for others. UCD focuses on the activities and processes taking place at the time of design, and largely ignores ‘support systems as living entities which can be evolved by their users’. UCD designers generate solutions for users who are relegated to a reactive role. PD involves users in ‘the process as co-designers by empowering them to propose and generate design alternatives themselves’. PD implies actively pursuing social inclusion and the active participation of users; however, this puts a strain on the system, which as to evolve to suit new needs (what I term reprogrammability). The system has to account for changing tasks and needs in order to incorporate new technologies. It is here that Fischer suggests the application of meta-design as a way to bring in users as co-designers of platforms, not only from the first, but for the whole of the platform’s life. Underdesign is the strategy or method taken from meta-design that ensures that users will be genuine co-creators of the platform.

I have experimented with both action research and activist research. They differ slightly in terms of who to involve as active stakeholders in the design process. According to Stringer (2014, 6) action research's 'primary purpose is to serve as a practical tool for developing solutions to problems experienced by stakeholders in the context'. In Stringer's eyes, it is devoted to helping people in finding solutions to their problems, and the researcher is there to provide them with a formula—that he has tested multiple times—to gain control of their situation and arrive at the optimal outcome. Activist research (Hale 2001), meanwhile, aims to better understand the root causes of the issue in question, done in collaboration with people who are subject to those conditions; it is used to formulate strategies for transforming those conditions and achieving the power to make the changes effective. Others in the activist research community have argued that neither action nor activism are necessarily participatory (Chatterton et al. 2007), looking instead to participatory action (or activist) research (PAR), but for the purposes of this thesis, I hold both action research and activist research to be participatory in nature.

Comparing Stringer's and Hayles' research methodologies, we can find slight differences in how they address the issue of participation in the design process. Action research, as presented by Stringer (2014), engages those who might help solve the end user's issues, but not necessarily the end users themselves. In that sense, action research covers UCD, but not necessarily PD. Activist research, on the other hand, works directly with the end users by making them the vehicle of change, a method that is closer to PD. As a platform researcher, both methods are needed: activist research to get closer to the end users, action research to get closer to those behind the platform's governance model (developers and promoters of the platform, mostly). This is the methodological approach I took during the SandS project. I learnt to separate 'end users'—an imaginary group who would have installed the SandS system in their kitchens—from 'project users'—the other researchers who used the prototypes to run tests, simulate scenarios, and develop their parts of the project. Given that the project users were also the main designers defining the infrastructure's usage and governance model, my role became that of action researcher, engaging with them to figure out the best way

to create a platform aimed at third parties. My design report on the SandS project (Cuartielles 2014a) included in the compilation here was created in collaboration with the project users, but mainly was addressed to them. The design we created—the ecosystem made up of the SandS motherboard and its module boards—required a manual for project users to use in approaching end users when working on their appliances and connecting them to the SandS cloud (Cuartielles & Taylor 2013b). We created this system with the intermediaries (the project users) between us and the end users in mind. The concept of the intermediary, as presented in Chapter 2, is taken from Callon (1991) as project users come with a baggage—‘the skills, the knowledge and the know-how they incorporate’ (135). The PELARS project was no different to SandS in this regard (Cuartielles et al. 2015). I had to design an electronics prototyping platform that would allow others to run experiments with end users. Those ‘others’ were my partners in the research process, just as in the SandS project; it was the project users and not the end users who were ‘my’ users.

By the time the SandS project started in 2012 I had already been part of the Arduino project for over seven years. While, at the time, I thought that I was engaged in action research, because of the interaction with thousands of end users on the Arduino Forum and via email, I later realized that what I was doing was far more like activist research, not so much because of the aforementioned difference in approach to user participation, but because of the political agenda behind the openness in design that we gave the platform (Löwgren & Reimer 2013; Cuartielles et al. 2018). Activist research (Hale 2001) aims to develop a third category of research, a hybrid one called ‘user-oriented basic research’ where the researcher does not set aside her personal beliefs. There is objective reflection on the facts, but not at the cost of sacrificing one’s worldview. Activist research brings an ‘additional demand for empirical rigor’ and a ‘well developed methodological canon’ (14) because of the way researchers become involved in the research process. In activist research, the researchers have to identify their deepest ethical-political convictions and use them to formulate their research objectives. Prior to establishing the research question, researchers have to engage in a dialogue with the users to be studied. As a matter of fact, this dialogue will continue

during the development of the project, and the ‘activist scholars’ will have to give it special priority. The research design must entertain the possibility of people in the project having different views, and there is the partnership with existing organizations to bear in mind—the research objectives have to coincide either partly or entirely with what stakeholders think is relevant to know.

An open dialogue with users is a key aspect of my research. For years I interacted on a daily basis in the Arduino Forum, a part of the platform structured as an open discussion list with topics determined by end users, and moderated by a subset of those end users. As an example of the interactions in the forum, there is the ‘Leonardo vs Vinciduino’ controversy in 2011–2012 (Arduino Forum 2011, 2012a, 2012b).⁵⁰ This began when the Arduino developers wrongly released a version of the Arduino IDE—the software used to program Arduino boards—which included information on a yet-to-be released hardware board. Armed with that information, a radically open collaborative process resulted in a board christened Vinciduino, which was somehow compatible with the still-unreleased Arduino Leonardo board.⁵¹ When we did finally release the IDE and the Arduino Leonardo board, the few changes we had introduced made the community-made Vinciduino board incompatible. This was a case when we brought our expert knowledge to bear, acting in what we thought was in everyone’s best interests—and taking the role of a benevolent dictator of sorts.⁵² The power of the Arduino actor–network was manifested by the release of reference designs that did—or did not—include others’ suggestions, but also by moderating the conversation, mostly because dialogue no longer functioned among the parties involved. It is almost impossible to be engaged in such emotional situations and keep a completely objective view of the process, and this is where activist research as a method gives me the framework to participate, as much as I need to grasp the fundamental dynamics of the platform.

50 The Arduino Forum at times keeps discussion threads open for years.

51 Arduino Leonardo is one of the so-called classic Arduino boards, running on 8-bit microcontrollers.

52 A benevolent dictator (see Chapter 5) has the power of the community to make decisions in case of conflict, but in principle will act only in the best interests of the group. In Arduino, technical decisions were taken in an assembly comprising the five founders.

For reference and to give a sense of why being so engaged in the process is relevant from a methodological perspective, it should be noted that we built the Arduino Forum to give anyone a voice in the process of building the platform. The platform rules, a standard set of terms and conditions, were augmented by Nick Gammon, one of Arduino's most active forum members with over 28,000 interactions in the forum at time of writing. Gammon published a series of posts in the same discussion thread, including explanations on how to use the forum, but also instructions on how to behave as a participant (Gammon 2013). This is an example of how to create a governance model where developers and end users share responsibilities and together build a platform that accommodates user's needs. While developers produce the technology, users decide the interaction at the social level. The Arduino Forum, which currently has more than 580,000 registered users, active members in 9 language groups, and more than 3.5 million posts (Arduino Forum 2017), is an example of a user-governed platform.

Other ways to involve users in a long-term relationship with a platform, as Fischer says, include underdesign, or making sure developers create opportunities for community members not only to co-design the platforms, but to continue participating in after the initial launch (Fischer 2003). Even if back in the 1970s the term underdesign had yet to be defined, the project 'Autoprogettazione?' by the Italian designer Enzo Mari was a good example of such an approach. Mari (1974) created a furniture exhibition catalogue for people to build by themselves with a very limited toolbox (hammer and nails) and wooden planks of just one width. Mari wanted users to become active participants in the design process. The one thing Mari asked for was for photographs of the furniture that people built from the blueprints. This call for feedback was the seventies' version of Gaver's probes (Gaver et al. 1999; Gaver et al. 2004). There were plenty of similarities in the formal aspects: the audience was limited, and thus clearly targeted, they were given a probe of sorts, and they were asked for something back, and they could choose whether to answer or not. Mari's blueprints differed from Gaver et al.'s probes in intentionality and expected outcome. While Mari's blueprints were intended to draw the user into the process as a participant, Gaver et

al.'s probes are just a way to collect insights from the user's everyday life. While the probes put the users in the position of continuing to build everyday objects, Mari underdesigned from the first.

What is left for designers to do in processes such as these? Should we teach future designers how to underdesign instead of how to design? There is a tension between underdesigning and creating scalable prototyping situations that involve technology, mainly if users need to acquire certain skills to become active participants in the co-creation process. How are users supposed to collaborate in the creation of a platform that has yet to be built? Similarly, are there any specific skills that designers should acquire when designing platforms? I will therefore turn to the idea of the designer as expert, with expertise obtained in multiple co-design processes in interaction with the platforms' various actors.

Emerging as expert

The role of the expert when approaching a case should be one of 'research facilitator, working with and supporting people to engage in systematic investigation'; pragmatic work for the action researcher, intent on achieving 'effective action' (Stringer 2014, ch. 1) and successful projects which are the result of a negotiated process between users and designers, who are sometimes equally active recipients of the action. Stringer explains that heeding the users' discourse will 'unleash energy, stimulate creativity, instil pride, build commitment, prompt the taking of responsibility, and evoke a sense of investment and ownership'. There is a very simple logic to such a way of thinking: an empowered user can be very useful in reaching a successful conclusion to a project. Stringer links this to Habermass's 'communicative action'. Dealing with open-ended research in a design project such as PELARS, when there is only a limited amount of time to produce it, requires decisions about when it is worth applying Stringer's approach and when not. My role both in SandS and PELARS was to design the underlying technology to be used by others in performing user tests. The way we approached participation was to produce a proof of concept prototype to be tested by project users, in order to determine how best to try out the systems with end users.

When it comes to the creation of electronic platforms with certain features like the ones presented in the PELARS project (Cuartielles et al. 2015; Spikol et al. 2015), there were few people in the world who could be listed as experts. Sections of the research community associated with the Tangible and Embodied Interaction (TEI) or Computer Human Interaction (CHI) conferences might have an understanding of the nuances of systems similar to the PELARS electronics prototyping platform, but cross-referencing that group with researchers who have experience of scaling-up projects to hundreds or thousands of users and the result is a handful of people. This was the situation when I joined the PELARS project. I was positioned as the expert, and as such I anticipated some of the decisions taken by the other researchers in order to ensure the success of the project, largely by having a fully functional and deployable prototype that would be tested by users in more than 100 cases.

Given my experience from the SandS project, which was in its closing phases just as PELARS started, I could anticipate many the potential technical issues with the expected affordances (by the project users) of the PELARS system. As the time we had to spend on the prototype was very limited—typically EU research projects have to be produced and tested in 36 months—I applied my expert knowledge to shortcut some of the technical challenges behind the design. This decision came at a price, for it is clear to me that my design decision influenced the outcome of the project from the moment it shaped the core aspects of the technology involved. My expert role in the design of the PELARS project was no different from that of co-founder of the Arduino platform back in 2005, when creating the first Arduino board. We had decades of accumulated experience in project-based teaching scenarios with hundreds of students, had helped with hundreds of projects, and therefore had a clear picture of what was needed in order to create a circuit board that could accommodate the needs of many of our students. This approach to the expert's role is different from the one described in the action research or activist research literature, which both demand constant contact with users in order to collaboratively create solutions to their issues. True, that contact did exist in the creation of the EU projects prototypes and the Arduino platform, but it required considerable technical effort before the

design was ready to be put in the hands of users to be iterated with their feedback. The novelty of the systems was one reason, as users had no idea it was possible to have an electronic system that could be reprogrammed for education purposes (Arduino), or a system that could be physically reconfigured and visually programmed to build design objects (PELARS). A proof of concept was needed in both cases just to start the conversation with users. Therefore, the role of the expert when dealing with projects involving new technologies calls for a slightly different approach, as the spell has to be broken, users need to see that things are feasible, and that technology can indeed provide solutions they never thought of, *before* starting the conversation about what is possible. To be told that something is possible is not the same as having it working in your hands.

Inclusive multiple prototyping

What, then, was my chosen method to take prototypes to large numbers of users? While prototypes are meant to be simulations of a final design (Rudd et al. 1996), that does not preclude them from being tested by large groups of people. A synchronous test of a technical or at least technically enhanced prototype would also imply a certain degree of scalability in technical terms. The method, which I came to call ‘inclusive multiple prototyping’ (IMP), hinges on the number of actors—part of an actor–network—involved in testing, using, or even learning from a platform. IMP is an ‘inventive method’ (Lury & Wakeford 2012), which implies using a device in a socio-technological network, where either the number of human actors (primarily users and developers) or non-human actors, or both, was large in comparison to the average IxD project. IMP forced me to set aside the more scholarly approaches of paper prototypes (Rettig 1994), provotypes (Mogensen 1992), or lo-fi prototypes (Rudd et al. 1996) that are widely used in interaction design. All of those approaches are based on small interaction patterns with users in order to gather a collection of cases in a valid RtD process (Gaver 2012), or on a proof of concept for a certain device or service that can be made quickly and cheaply as the focus of a conversation with end users about the opportunities offered by the technology. IMP is not just about technology, but about a combination of products,

services, and the different types of users involved. Its principle is to prototype a real-life scenario where new technologies and/or processes are involved.

The ‘inclusive’ element in IMP refers to the possibility of participation being open to anyone, and is again far more in line with the basic tenets of democratization—a central theme in this thesis. The ‘multiple’ element highlights the number of actors, but also the vertical coverage of the platform, by looking at which of the different layers of the sociotechnical protocols operationalize the actor–network and provides interfaces for them all. Finally, ‘prototype’ refers to the idea that contemporary design is never done. Even when we hold it in our hands, it is unfinished, albeit usable, right up to the next firmware upgrade. In a way, a platform—where long-term sustainability depends on a continuous co-design process—is a prototype. The IMP method calls for strategies to collect data in complex scenarios with hundreds of IoT devices in one network. In the cases I was involved with it was important to be able to reaching out to as many people as possible, which required a different approach not only in terms of the technical infrastructure of each project (which was still at the prototype stage, of course), but also in terms of the kind of data generated, the ways to deal with documentation, or the finances needed to put the project in motion. It is one of the reasons why I would argue that IMP is a research method of relevance.

Just to give some idea of how much size mattered in some of the experiments, in the PELARS project the end users were the teachers and students who would use the system in pilots assisted by a group of researchers. This demanded prototypes that were as finalized as possible, ready for systematic tests. I was responsible for the design, manufacture, and integration of the Talkoo prototyping kit that had to be fully integrated into the PELARS ecosystem (Cuartielles et al. 2015). In the first iteration alone the kit included 13 different circuit boards—and 50 copies of that first kit had to be produced for the project participants to experience how it might work and to develop educational activities with it. A later version of the kit had only 10 boards, but over 100 copies of that kit were needed for the various project partners, and also as a dissemination action for the project. As

a result, in order to conduct some of the project's pilot tests and some dissemination actions, we had to manufacture over 1,500 circuit boards that were distributed to over a hundred partners and potential end users. One of the partners, Malmö University, conducted over 100 experiments with groups of twenty or more users at a time. In engineering terms, when looking at the numbers, one might well ask whether we are still talking about prototypes or about a small production batch of a product.

When dealing with end-user software, where the number of application users can run to the thousands, this is definitely not a large number, but when dealing with hardware, and pilots with scores of designs manufactured in the hundreds for thousands of people, it reaches a different level of complexity. It is a lot more tedious to prepare the pilots, but still not as hard as it is to prepare the production of a mass-manufactured product. These inclusive multiple prototypes are still prototypes in the sense that they have not reached the market and are not publicly available for purchase, but they are more than simple prototypes used by single users in a more constrained test context. The PELARS prototypes are open in nature as they do not have a defined purpose, being the building blocks of a prototyping platform; this means there are some things we can expect from the users, such as certain types of projects we may have anticipated in our laboratories, but much of it will come from the interaction of end users with the actual devices.

Concluding reflections on mixed methods

This chapter has set out the methods I used throughout my studies when interacting with users, building prototypes for large series of case studies, extracting relevant information, or deciding my personal take on the projects I am engaged in. There is a difference between my approach and most of the literature on design, which focuses on small-scale cases where the design researcher has direct contact with users. I have constructed a discourse from the best practice of various practitioners in the arts and design in an attempt to explain how my process is not very different from theirs, except for the fact that the outreach of the experiments has sometimes been far larger

at my end, requiring a different kind of planning and the adoption of an expert role that might be thought contrary to the one suggested by the literature on participatory activist research. I would suggest that it is not in fact misguided, as the cases I have been involved in show, but also given the fact that technology in future will be made of complex systems with hundreds of data points, and where a different design mindset will be needed in order to achieve successful designs.

Having coined the term ‘inclusive multiple prototypes’ for preliminary designs that either have many parts interacting in a system, or that have to be tested with a large set of users (or both, as in the PELARS project), I would argue that it should be given serious consideration by interaction designers when dealing with new computing paradigms such as the Internet of things, that agglutinates ubiquitous computing with pervasive computing and requires designers not just to understand the underlying technology, but also the distributed networks and data flows. I have successfully pursued a series of projects that can be used as examples of a methodology that designers might find very useful; a methodology that can be summarized as high-speed, open, user-governed activist research, where much of the learning and contributing happens in direct interaction with users who are trialling the prototype platform. It is high speed because it is not possible to reach certain stages of development with a traditional ethnographic analysis of the work, for while it may gloss over some of the nuances of the design, at the same time it makes it possible for it to develop rapidly as a system.

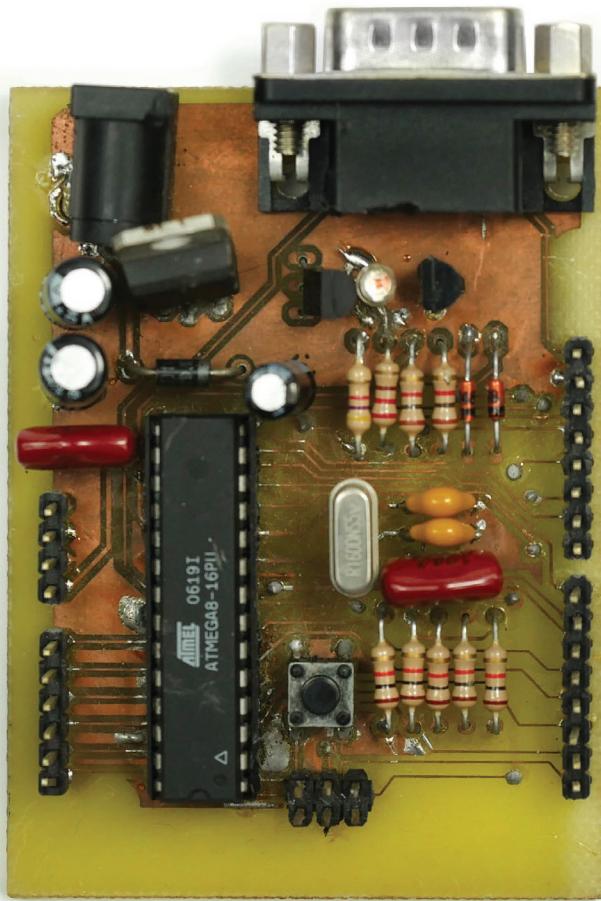


FIGURE 5
ARDUINO COMPATIBLE SERIAL BOARD
MADE BY A COMMUNITY MEMBER, 2006

5 LESSONS LEARNT

This chapter is a re-examination of the preconceptions outlined in Chapter 2 in the light of the experience of developing the projects described in this thesis. It became evident that there were intentions, needs, approaches, and mindsets that only transpired as the projects evolved, and since my original framework, the manifesto detailed in Chapter 4 (Cuartielles 2004), was never meant to be a hard and fast rule for how to proceed with the research, more a display of the possibilities, I simply incorporated new perspectives on how to deal with some of the as yet unsolved challenges in platform design as they emerged. The result, presented here, is a revised framework that takes an experience-based, augmentable, and modular view of platform design. No longer detained by spatial (size) and temporal (performance and pace) issues, this framework 2.0 is more holistic in its approach, incorporating the soft qualities that are central to contemporary design discourses.

Not only did my thoughts evolve as a result of my practical experience gained while co-creating systems such as Arduino, the indoor location system, SandS, PELARS, or the haptic-enabled wearables, but also because of I have closely followed the literature on interaction design, the sociology of technology, the arts, education, and engineering. Given that I have been engaged in this process for a longer period, I have had the chance to witness how the evolution of technology and its interaction with us humans has been disseminated in academic publications, the specialized media (such as engineering magazines), as well as non-specialized literature. The last fifteen years have seen the paradigms of ubiquitous and pervasive computing become a reality. The sociology of technology has returned to actor–network theory with renewed interest thanks to new studies that apply it to contemporary technologies such as the cloud, APIs, and AI agents (Bratton 2014, 2015). In the context of education, technology seems a catalyst in the move towards learning paradigms

such as project-based learning or experiential learning (Säljö 2010). It is now common to see interactive digital technology as part of artistic performances and integrated into artworks (Fry & Reas 2007, 501). Engineering is flirting with the new values generated by AI by addressing the ethical questions raised by autonomous vehicles moving freely on the streets (Johnson & Miller 2008; Levinson et al. 2011). Trade is conducted by automated systems (Cox 2017). Even insurance companies are looking at the potential risks and opportunities of the interactions between actors in the always evolving network of connected flesh, cogs, and bits (Yeomans 2014).

This double helix of praxis and theory has led my understanding of the field to drift away from the technology-centrism of my early work towards one where an assemblage of non-human and human actors share the spotlight. I have come to understand the ideas suggested by actor–network theory as a continuously constructed tracing of interaction patterns, modes of use, and social relations (mediated by technology). I myself have traced the actor–network of platform design through an iterative design process, which has brought into play a new set of assumptions to help mapping the networks in a way that looks beyond their spatial and temporal attributes. It is analogous to the microelectronics business—it is a complex endeavour to set up a factory dedicated to the production of microchips at the highest level of miniaturization in one take. The creation of such a megastructure (and I am talking here about the process of building an extremely precise assemblage, where humans and non-humans work in a never-ending shift and processes and artefacts have to be synchronized to inform one another in an unceasing cycle) requires an iterative process of refinement stretched out over a long period.⁵³ Reflecting on the relationship between these complex assemblages and platforms, I reconsidered which tools are needed for the creation of platforms.

⁵³ Before starting at Malmö University in 2000, I worked as microchip designer at Infineon in Munich, Germany. I had the opportunity to visit their Dresden facilities, a factory constructed on a field that was used as a tank cemetery at the end of the Second World War. The site was built in record time by thousands of builders working round the clock. Such a facility is always in operation to maximize the return on the investment needed to start it.

Thus this chapter presents the set of tools that have become part of my personal design toolbox, and which have been refined through an iterative design process. I would suggest they make a useful framework for designing and building platforms, as they bring together a set of discursive elements to open the conversation when co-creating a system. All the terms have featured in the thesis thus far—sustainability, obsolescence, openness, marketability, community, democracy (from technological empowerment), the political, ecology—and, rather than being intended as absolute truths, they are open for negotiation, being tools to help when modelling assemblages and their contexts and when dealing with new platform design. They are not all needed at once, and each might or might not be needed at a specific moment in the evolution of a platform. For simplicity, I present them in a logical sequence of sorts, but this is not to imply that they should be introduced into the design process in the same order, nor that they should remain unaltered throughout. Just as in the example of building a microchip factory, iterations and a continuous evaluation of the process are needed when building a platform—it is the price to pay for the platform's featurelessness.

In describing the terms I have continued with the chronological disposition as the most appropriate. Thus I begin by introducing the discursive tools in the order of the papers in the compilation, while the in-depth explanation of those terms must wait until the second part of this chapter, where I enlarge on the discussion by adding a critique of each of the topics.

Papers and other projects

This first section will look again at the papers in order (see Chapter 3), but also at the projects I have used to illustrate the various cases considered in this thesis. By looking at how the terms shifted in relevance over time, the process of understanding how platforms are formed becomes into focus.

Paper: Research design applied to platform creation

When I wrote my resign desearch manifesto in 2004, I was unaware that platforms would become increasingly relevant, and thus instead

of studying the design of platforms, I was more interested in the idea of how *digital manipulatives* as *digital materials* could become part of everyday design practices (Resnick et al. 1998; Redström 2001; Cuartielles 2004). The idea of *democracy* from the perspective of access to tools and processes (see Chapter 2) has always been my interest, and it is therefore the first of the terms to feature. In the manifesto, democracy is related to the concept of design collection as a way to reach as broad an audience as possible, and thus be more accessible. Closely related to it one finds the idea of *marketability*: given that Sweden (like all of Western Europe and a majority of countries worldwide) is a capitalist economy, and given that technology is materialized as products people can buy, including the idea that the design objects had to be attractive to contemporary economic cycles of offer and demand, marketability came very early to be part of my discourse. Many of the projects I realized between 2005 and 2012 were funded either privately or aimed to reach a certain degree of commercial exploitation. This has ethical implications that I will discuss later when exploring the political aspects of platforms in greater depth.

Project: 'Micromobility and Learning'

The first research project I joined, 'Micromobility and Learning', looked at how education related to learning spaces (described in Chapter 1). From this project I learnt how to describe an assemblage using the term *ecology*. It was when considering educational experiments that the conversation turned to the fieldwork methodology used to collect data in experimental settings. The configuration of a classroom is typically determined by certain conditions—its location, the number of teachers, the number of pupils, the duration of the class—which put together comprise the class's ecology. When running an experiment about the use of a specific technology in a classroom, the nature of this ecology is transformed by the temporary presence of exceptional actors: researchers, cameras set to record whatever events are thought relevant, people take notes and ask the participants (pupils and teachers) about how things feel or work. This gives the class a different setting, a different *ecology*.

An analysis of the ecology of an experiment matters because of the question whether the experiment could become the norm after the researcher's intervention. It may be a challenge for the teacher to repeat the same kind of experience without external help. The question is then whether the experimental ecology design was good enough to make the whole assemblage sustainable. *Sustainability* here refers to the resources needed to support a certain situation. Other possible meanings of sustainability—namely those referring to environmental aspects—did not become part of my toolbox until later.

Paper: An indoor location system that went nowhere

Research projects can have more than one outcome. The indoor location system produced as part of the research done with the University of Zaragoza—and which became sideline in the ‘Mobility and Learning’ project—had an excellent academic outcome.⁵⁴ Our work was published in *IEEE Pervasive Computing* (Casas et al. 2007) as a paper that looked not only at the technology, but also at the hidden issues behind installing something as complex as an indoor location system. Among the learnt lessons included in the paper were a great many interaction design aspects, such as how to configure a system with the smallest possible number of operations, which kind of installations could take place, and how such a system (conceived at the beginning of the IoT era, after all) could be implemented in spaces that were never designed to host embedded electronics in their structure.⁵⁵

While the technological achievement of the project was remarkable, it was less clear how to proceed to make it useful beyond its academic results.⁵⁶ When the time came for us to decide how to ensure the inven-

⁵⁴ The indoor location system, which went by the name of PISHA (Positioning Indoor System of High Accuracy), was used in one of the ‘Micromobility and Learning’ experiments to track people’s movements in the open spaces at the School of Arts and Communication.

⁵⁵ In 2003, rather than the IoT we used to talk about WSN—wireless sensor networks—which preceded the IoT by several years. From the beginning the indoor location system was meant to be a special instantiation of a WSN, a network where the various active moving wireless devices could be located by other wireless static devices.

⁵⁶ We had in our hands a tool that could localize objects in the three dimensions of space, at a refresh rate of 10 times a second and with an accuracy of 5 centimetres, using the equivalent of a satellite system such as GPS or Galileo, but for indoor use.

tion continued, there was no agreement on how to move forward. It being 2003/2004 the concept of open-source hardware licences was still unfamiliar at the institutional level.⁵⁷ Our only option seemed to be to patent the technology and try to sell it on, or even to start our own company to exploit the results. The main problem was that our technology was a proof of concept, and it was a good ten years ahead of its time.⁵⁸ So while the indoor location system was very interesting experiment, it was not shelf-ready technology waiting to be adopted by a start-up company as its main source of revenue. In the absence of dissemination alternatives, compounded by differences within the team on how to proceed, the indoor location system was doomed to disappear as a technology.

One term that arose in discussion during the final phase of this project was *openness*. In interacting with my partners at the University of Zaragoza, I learnt that openness was an activism of sorts. Why else should one create something and share it with others with no apparent economic benefit attached to it? It would be years before open-source hardware would be recognized as an accountable outcome for research.⁵⁹ When it comes to business, open source and open innovation (a more abstract term) have been reported to produce indirect outcomes for those generating the technology (Hippel 2005; Cuartielles et al. 2018).

Project: Arduino

The Arduino project plays a pivotal role in my research. With a community of hundreds of thousands of users participating on its online forum and millions of visitors every month (Cuartielles et al. 2018), Arduino has transformed the way people understand educational technology by moving from the institutional laboratory to the personal, portable one. It has democratized access to the exact

⁵⁷ Open source hardware did not count in the legal framework in 2005, when we were all but finished working actively with the indoor location system; now it does count, making it harder to find support for the project among institutions or companies.

⁵⁸ At time of writing, some companies are starting to offer services using indoor maps for public spaces and the like, while technological developments, mainly in Ultra-Wideband, make indoor positioning small enough to be used in all sorts of everyday applications.

⁵⁹ The first open hardware journal was HardwareX by Elsevier (<https://www.journals.elsevier.com/hardwarex/>) and did not appear until 2015.

same tools for people from all over the world at prices adjusted to their needs (thanks to the arrival of derivative designs and clones on the market). Arduino represents many things, and it definitely includes—as a platform—all of the terms I present as tools in this chapter; however, the most important is the idea of *community*.

Building Arduino forced me and my partners on this journey to think beyond the technology. All of the sudden, our work was relevant to people on the opposite side of the world in ways we could never have imagined.⁶⁰ Arduino is used for learning about digital technology, building electronic controls for aseptic rooms, launching small low-orbit satellites, making machines to explore the bottom of the ocean, controlling liquid mixers in the food industry, or building braille learning devices for children with visual impairment. The sheer number of people using Arduino in their daily lives makes it very successful as a project. Many of them collaborate on the technology's further development, assist others in the online forum, arrange real-life events to talk about it, or write books to share knowledge about the platform. These people comprise the Arduino community of users and developers. No other project I have been involved in measures up to Arduino in terms of community-building.

Paper: Haptics

The various projects on haptics dealt with the idea of spatial and/or temporal interactive experiences. The prototypes analysed in the papers were built to explore the possibility of creating multiple artefacts that could interact with a person or a collective because of either spatial or temporal properties. All of the experiments investigated the concept of inclusive multiple prototypes, introduced earlier (Chapter 4). We jumped from an experiment where scores of actuators attached to the body were triggered in preprogrammed patterns according to the location of the user, to one where hundreds of actuators on one suit were activated by another user touching an identical suit at a

⁶⁰ Not only did we not anticipate the success of Arduino, we did not foresee its appeal for people who were not our intended primary users. For some people, Arduino has become their main source of revenue, as I explained in my essay 'The Power of the Copy of the Copy' (Cuartielles 2011); for others it was the path to a career in engineering; and there are people making machines to improve their daily lives by automating simple tasks, etc.

remote location. All the prototypes were extremely time-consuming to build and needed every hardware and software trick in the book to make them work.

If there was one word that dominated in this project in a way not seen in the others, I would say it was *obsolescence*. It was centred on the key limitation while building inclusive multiple prototypes: money. To build the prototypes, hundreds of circuit boards with processors had to be made. At the time, we could not afford expensive prototyping services or parts, and this forced us to be very creative in the way built everything. We made revived a whole series of obsolete techniques, including token-ring to implement the communication between the various parts of the prototypes.⁶¹ But the term obsolescence has a much deeper meaning, especially *planned obsolescence*, which can be contrasted to the idea of *enough computing power* which I presented in the theoretical framework of the thesis (Chapter 2). I will detail the concept below, along with the relationship of obsolescence with sustainability.

Project: 'Creative technologies in the classroom'

It was not long after we built Arduino to give our students access to state-of-the-art embedded hardware development tools for interaction design that the tools themselves were adopted wholesale by other educators. As well as redeveloping the Arduino tools as a platform, I joined a group of upper secondary school teachers in experimenting with the adoption of embedded technology in the standard school curriculum in science and technology. I conducted workshops on basic twenty-first-century digital skills and computational thinking for educators in Spain in 2006 and Argentina in 2007 and 2008; workshops that resulted in the creation of educational resources by the teachers themselves, designed to be used in their classes. I was concerned about transcribing my process so it could be adopted by

⁶¹ Token-ring is a networking technique whereby all the devices are arranged in a circular configuration. Each device can only communicate with the one on the left or the one on the right. In order to send a message to any device in the network, the information packages have to travel through a sequence of other devices to the receiver. Communication is only one-way, so that even if a device could technically communicate in two directions, it will only do so in one. The name 'token-ring' comes from the circular shape of the network and from the fact that, in order to determine which device has the right to send data, the devices pass around a digital token that gives them the right to occupy the channel.

others, but it was not until 2012 that I had the opportunity to design a project to address some of the issues I identified during those early experiments.

The ‘Creative Technologies in the Classroom’ (CTC) project thus represents my personal take on a formal education in technology at the upper secondary level. It was intended to become a platform where teachers could learn about educational technologies while teaching. *Sustainability* was the key concept, as the intention was to help teachers become self-sufficient in exploring the use of digital technologies in their classes. CTC was (and still is at the time of writing) a project with a political agenda, devised to return power to teachers so they can decide what is relevant in an ever-changing context; digital literacy is simply not enough, for so-called twenty-first-century skills must be constantly refreshed, as digital technology is a fast-evolving field. The project not only began by centring on an inclusive multiple prototype—with hundreds of people involved in the first iteration, jumping to thousands of users for the second and all subsequent iterations—but it continued to do so, as I managed to run the process internationally over a period of four years. Marketability became increasingly relevant for this project, as the NGOs covering its expenses could sometimes not provide enough materials for the schools, necessitating that we make CTC kits into products to help the schools otherwise left without the parts they needed.⁶²

The experience of using a kit to achieve something technically challenging is an empowering one. I have seen it in a range of projects, resulting from different instalments of CTC. Now in its fourth edition, CTC is still a kit used to show upper secondary teachers how to introduce technology to their classes. The project has been run in Spain, Ecuador, Mexico, and Sweden, and by the end of 2017 had reached over 17,000 pupils in interaction with more than 1,550 teachers in over 730 schools, using 860 kits. Given the nature of the project, the kit played a very important role, and its design was

62 ‘Creative Technologies in the Classroom’, which brings technology teachers up to speed with contemporary educational tools, is a project I designed for the Arduino company, sponsored by the foundations ‘Fundacion Telefónica’ and ‘Fundación Bancaria La Caixa’ in 2013, and an official Arduino product as of 2018.

iterated year on year to include feedback from participants. Besides a few bugs in the software or the occasional problem with the online documentation system, the response from teachers and pupils was generally very positive. Thanks to the interaction with the kit and with other teachers in the project, teachers were able to pursue a project-based learning methodology that included the use of software and hardware tools, all of them documented as interacting with one another as part of the kit.

Paper: SandS and PELARS

The two European research projects I have included in this compilation, SandS and PELARS, are represented by my design diaries for projects. Each paper analyses the work of creating an accessible technological platform to connect the physical world with a digital artefact, and from there to other artefacts via the Internet. The inclusive, multiple nature of the two projects comes from their coverage of an entire vertical segment of their respective fields (IoT for households in the case of SandS, computer-assisted super-fast prototyping with digital tools in the case of PELARS). Both projects had a series of prototypes, user tests, and long development times, and resulted in artefacts created with a degree of marketability in mind.⁶³

Besides looking at the possibility of the research results becoming products and services available to a larger audience, SandS explored the idea of obsolescence by upcycling existing non-connected home appliances to a dedicated cloud so that users could use them for new types of task. PELARS, meanwhile, investigated the democratization of prototyping tools by creating a more egalitarian system, which, thanks to its ease of use, would make digital prototyping with physical artefacts possible for people who have no knowledge of software or hardware.⁶⁴

⁶³ The SandS project paved the way for the creation of the Arduino Yun board, which sold well because of its innovative approach to offer connectivity to the Internet as well as low-level control of physical drivers for latency sensitive devices such as motors. For the Arduino Yun, Arduino was awarded the Innovation Luminary Award by the European Commission in 2017.

⁶⁴ The PELARS toolkit is still being used for user trials by researchers at Malmö University, long after the final deliverable of the project. It is as yet unclear whether this design will become a product. Arduino tried to speed up the process by launching a crowdfunding campaign that did not go ahead

Overall concepts

Going back to the series of spatial preconceptions (tool, toolbox, kit, platform, and infrastructure) I presented in Chapter 2, because they only concern the size of the system they barely scratch the surface of platform-building. To recap, a tool is about a person's interactions within a context via a machine; a toolbox involves several machines and materials; a kit introduces in the notion of a predetermined goal, plus the intention of educating the user; and a platform and infrastructure take it to the level of public and private. When it comes to the temporal aspects, performance concerns computational power and processing speed. An analysis of a system using such terms will be very techno-deterministic in nature. While they offer a valid view of where platforms fit in a larger ecosystem of things, I have observed over the course of the projects discussed in this thesis that an entirely different terminology is needed specifically for building platforms. A simple analysis in terms of scale and performance cannot explain the various aspects my experiments show to be relevant. This section thus offers a whole new series of functional requirements (to borrow a technical term from computer science) for designers to approach platform design with new eyes. The concepts are arranged in a progression from technocentric to the participants' co-learning, via collaborative action in the creation of platforms.

Sustainability

Krippendorff (2005), Callon (1991), Latour (1996), and Bennett (2005) help in describing the idea of an assembly as ecology (which is discussed below), there are practical frameworks that could serve as examples of how it has been applied in real life. One documented case is USAID's document referring to the 5Rs framework (USAID 2016), which promotes sustainable projects and activities via local actors and systems. The 5Rs framework takes its name from its approach: the ecology of systems should comprise the five R's of results, roles, relationships, rules, and resources. While results represent an intangible, unexpected, and unknown asset (we cannot foresee the future), the other four terms are simply another way of looking at actors

for reasons that were purely financial—the sum needed for the project to be viable was larger than the community was ready to support.

and intermediaries (Callon 1991), and how they interact in building networks. Sustainability, which is the term under discussion here, depends on realizing results that stakeholders truly value, and the ability of the system to produce valued results over time (USAID 2016)—in other words, it is strongly linked to the idea of duration over time, which is achieved if the results are satisfactory to the people involved in the process.⁶⁵

Sustainability can be also understood from the point of view of availability. For example, if a certain technical resource is simplified in order to make it easier to replicate, more of that resource can exist, as Raghavan and Hasan (2012) conclude when looking at Internet infrastructure. They conclude that a system such as the Internet, which can be implemented so that it is free of single points of failure by simply replicating pieces of the technical infrastructure, is far more sustainable thanks to having readily replicable, and thus readily available, devices, which seems to also be more environmentally sustainable. Raghavan, teamed up with Barath and Pargman, offers a closer analysis of this ecological aspect when studying ‘sustainable’ human–computer interaction (Raghavan et al. 2017), and suggest that intermediation is responsible for more complex and therefore less sustainable resources, and ‘increasing complexity in itself is unsustainable’. Their work is not a manifesto for simple or simpler systems, but for the removal of intermediaries that increase the number of steps to the outputs of systems, and with it the cost. An example of this cost increase is the one mentioned earlier when talking about openness and the patent system in Hippel’s work (2005, 114), where costs increase as more patents from different sources are applied to the same product, ultimately making it unsustainable. But Hippel, just like Papanek (2011), Mari (2017), and Lessig (2004), is more concerned with how openness in design can make design more sustainable. If someone figures out a way to improve a certain design which they then publish, it will be more widely used, easier to manufacture, and by extension more sustainable. Hence the extended life of GSM technology, the communication technology that powered the

⁶⁵ The metrics of how to measure satisfaction vary depending on the context and circumstances. For developers, satisfaction comes from their work being appreciated by users; for users, if the system works as expected; for all, if a system functions as long as it is needed.

first digital mobile phones, which is still in use as a fallback solution and for the production of cheap mobile devices. As the patents behind GSM slowly expire and therefore become ‘open’ (Goodman & Myers 2007), the added expense of licensing over and above the technology vanishes, just at the time when the increased knowledge about how to manufacture better modems using the GSM protocol makes it less complex to produce communication devices using that technology.⁶⁶ Thus, this technology is more sustainable, while being more resource hungry—less environmentally sustainable—than other more modern technologies. As technology evolves, communication protocols use less bandwidth, less transmission power, higher compression rates that save power and transmission spectrums, thus making newer devices less resource-hungry, and that translates into needing smaller batteries and therefore being less harmful to the environment. If we look back at the history of radio communication, the simplest AM radio receiver can be made with a potato as a power source and a few components. It is hard to get rid of older, largely deployed technologies for two reasons: users may resist acquiring new devices to perform the same function as the older ones, and these technologies are—by definition—more sustainable as they are technically simpler (this is only temporary, though, for once the new technology is widely adopted, it is the old one that becomes less sustainable).

Marketability

Marketability is a concept directly linked to the economic sustainability of a platform, and while it might be of vital importance for its raising and maintenance, not all platforms are necessarily built on a market paradigm. For platforms there is a strong link between sustainability and marketability. Marketability determines whether platforms will last over time, thanks to their ability to generate an economic flow that means they can survive and grow as part of the capitalist system they belong to. When building a platform, people become invested in the process because of very basic interests such as the need for tools to execute certain actions or social acknowledgement as documented in the analysis of open-source projects by

⁶⁶ Based on this premise, various vendors are still manufacturing boards including GSM modems—for example, Arduino launched its MKR GSM board in 2017 to leverage GSM connectivity for as long as it exists, which could be several years.

various studies (Malinen et al. 2010; Geipel et al. 2014). However, the creation of large platforms seems to call for mixed models—what Hippel (2005, 91) calls a ‘private–collective’ model of innovation incentives—in which the various people agree to have different interests, some of them being economic in nature, as seen, for example, in the case of Guifi.net in Spain (Baig et al. 2016) or Arduino (Cuartielles et al. 2018). What Guifi.net and Arduino indicate is that the long-term sustainability of a platform is going to depend on the ability of its members to build a relationship with the existing economic system, which for Western Europe at time of writing is the market economy. This relationship has to be satisfactory to all parties—users, developers, and any intermediaries—and it has to be formalized with an agreement where the intentions of all actors are clearly described.

At first glance, the evidence seems mixed as to whether it is only in a situation such as Guifi.net’s or Arduino’s that a hybrid system can be sustained, in which some of the collaborators are volunteers and others receive recompense for their participation. There are other platforms, like the one supporting the creation of the kernel (the core, or the basic building block) of the Debian operating system, where all of the work is supposedly done by volunteers (Zacchiroli 2011). In my opinion, Zacchiroli, in his lecture of 2011 (on the topic of the creation and maintenance of the Debian OS) stated that the Debian developers were not paid by Debian, but *not* that they were volunteers. Consider the *Linux Kernel Development Report* from the Linux Foundation (Corbet et al. 2012), which said that 75 per cent of the Debian developers were paid: ‘7,800 individual developers from almost 800 different companies have contributed to the kernel ... in fact, the individual development community has doubled in the last three years’. Its 2017 report revised the figures upwards: ‘15,600 individual developers from over 1,400 different companies ... The number of companies supporting work on the kernel appears to be stable and not growing like the number of developers [...] well over 85 per cent of all kernel development is demonstrably done by developers who are being paid for their work’ (Corbet & Kroah-Hartman 2017). What this demonstrates is that, in order to create a sustainable platform, it is important to be on a good footing in the existing socioeconomic paradigm. In Linux’s case, the market

contributed 85 per cent of its development costs in the form of corporations paying for the development time by getting some of their employees to write code that is ‘pushed’ to the common pool of code.⁶⁷ For example, according to the report, the largest contributor to the Linux kernel in 2017 was Intel, one of the largest corporations in the world. The true volunteers Zacchiroli was talking about only account for some 15 per cent of contributions; however, this has doubtless changed over time, as the evolution of the reports shows. Going back to 1991, when Linus Torvalds, creator of Linux, published his work on the invention of the Linux kernel, 100 per cent of the contributions were made by volunteers, but as the importance of Debian has grown over time, other actors—including some with a financial interest—have become contributors. Yet without the volunteers, the platform would not exist. The conclusion to be drawn from this is that relationships between human actors and the platform change over time.

It should be noted that marketability is not opposed to openness, as the Debian case shows. By making knowledge freely available, it does not mean that the work needed to produce it should be done at no expense, nor that the knowledge itself cannot be sold. To quote Stallman, ‘*Free software* is a matter of liberty, not price’ (2002, 43). Later in the same essay, Stallman develops this argument by identifying three layers to the marketability of software: creation (development), use, and distribution (intermediation).⁶⁸ These three define the current business models for the creation of open knowledge, allowing developers to make a living by creating better tools, and thus making the system sustainable. For example, in the case of my own experiments, CTC was an open educational hybrid of service and product that was developed with funds from private foundations, and is now sold by the Arduino company, while Arduino boards are

⁶⁷ Push, pull, commit, add: all are common terms in the software world, being the basic commands for sharing code in Git, the open-source software used to co-create software applications with the collaboration of thousands of contributors. ‘Push’ refers to the command to send one’s code to the shared repository.

⁶⁸ Stallman 2002, 44: ‘Thus, you may have paid money to get copies of free software, or you may have obtained copies at no charge. But regardless of how you got your copies, you always have the freedom to copy and change the software, even to sell copies. Free software does not mean non-commercial. A free program must be available for commercial use, commercial development, and commercial distribution. Commercial development of free software is no longer unusual; such free commercial software is very important.’

open designs that run on free software, but they are sold on both online and offline stores all over the world.

This offers an opportunity to discuss the ethical and political implications of working with a project like Arduino, which evolved from open-source project to incorporation as a company in several countries. Being part of a project like this, as both a participatory activist researcher and co-founder, raises questions regarding the ability of the researcher to remain true to the participatory aspects of the project. Marketability implies being part of a market, where people buy and sell products and services, where decisions are not always made based on the same values as those of participatory activist research, and where the researcher is just one part of the governance mechanism and not the one with the last say. Marketability, as a characteristic of the platform, can conflict with its other values. This is yet another reason to take the market into account and be ready to consider how much it could compromise the projects functional requirements and intentions. In the case of the Arduino company, to follow up on an example mentioned previously (and always judging in retrospect), the radical openness of its development processes could have hampered its decision-making, which would have impacted on the marketability of the tool and therefore its sustainability. The question remains whether we could have done things differently, while still ensuring the sustainability of the project. Was optimizing marketability the way we did it a good-enough strategy? To what extent did concerns about marketability compromise the establishment of the Arduino community? When looking at the current metrics of the Arduino project in terms of usage and Internet popularity, compromise might not be the first thing that springs to mind; however, there is little to be said about things that never happened, and we can only go on the data that resulted from the actions we took. The rest is the outcome of a lucubration exercise that is beyond the scope of this thesis. In short, there is no secret formula for ensuring that platforms are economically sustainable; the three cases documented in this section are different in nature, yet all seem to work and are generating their own actor–networks. Plainly there are different models for how platforms, open or not, can become economically sustainable over time.

Obsolescence

We saw earlier in this section that sustainability as a concept is linked to obsolescence when looking at the longevity of a platform. In contemporary technology design, we are confronted by the concept of *planned or programmed obsolescence*—in other words, the possibility of deliberately programming the disappearance of a system (London 1932). Both products and processes can become obsolete on the arrival of something better, although the new does not necessarily eliminate the old from the context of use: both the carpet sweeper and the vacuum cleaner continue to co-exist in contemporary households, even if the latter made the former obsolete (Nelson 1967). There are various reasons for this. There may be government-policed technical obsolescence programmed into infrastructure and services in order to increase their capability, the classic example being the migration of analogue television to digital terrestrial television. In recent years, analogue has been wound up in all European countries (and many others), replaced by the far more spectrum-efficient digital terrestrial television. This change was programmed by the different countries and policed through their institutions—public or private—dedicated to the management and supervision of the telecommunications' spectrum. In past years we have seen this happen for many different technologies. Think only of the migration of 2G telephony to 3G, and later 4G, and soon 5G. The way this is enforced is typically by increasing taxes on the services that use the old systems; what forces the enterprises offering the services to switch to the newer technology also encourages users to migrate. Some have argued that this degree of control, limiting obsolescence just to keep the existing business models running, is counterproductive (Lessig 2001, 83). Nor is programmed obsolescence always immediately successful, as seen in the case of GSM communication. This, as Nelson says, is ‘a waste’, and not always a planned process at that. Indeed, obsolescence can happen accidentally, much as the car made the traditional city obsolete, because its streets, parking, and buildings were not designed for the changes imposed on architecture by its arrival.

Obsolescence is a consequence of the techno-economic development of a culture (Nelson 1967) and is wholly dependent on the historical moment any given region is in. After the Second World War it was

standard practice the US, for example, when the country was capable of producing and selling consumer products in huge quantities while still largely investing in capital goods. Yet the term ‘planned obsolescence’ comes from London (1932), and an essay in which he claimed it could be used to keep production and consumption buoyant (London 1932). As Nelson (1967) writes, obsolescence may not just be a consequence of technical advances; it can be purely aesthetic. He uses the car industry as an example, where users end up accepting a financial loss by purchasing the new models that appear every year. Objects are obsolete from the moment they are purchased, sometimes from the moment they are designed, and, in some cases, even before the process of patenting an invention is complete (Ackermann 2009, 195).

The philosopher Herbert Marcuse, with his critical view of capitalism, takes a pessimistic line on obsolescence, explaining that it is part of the system’s design and not an aberration (1969). He argues that its *raison d’être* is the perpetuation of the ‘struggle for existence’. He was investigating the idea of tolerance at the time, and referred to obsolescence to show it cannot be considered a bad thing as long as it is tolerated by consumers and producers. At the same time, tolerance compels the consumer to get used to change and to plan for that change. Lessig (2001), looking at the issue from a more liberal standpoint, implies that planning for change is a handbrake on true innovation, for he suggests that anything that forces technology to operate with a planned obsolescence paradigm (as is the case with communication spectrum policing) limits the chance of revolutionary technological innovation (85). Lessig toys with the idea of removing not only governmental regulation of the spectrum, but also market controls, although he acknowledges that it might rapidly lead to the tragedy of the commons.

There is another point to be made about obsolescence, which is the extent it is determined by punctualization. There are systems that do not necessarily become obsolete because of aging technology, but because, having become ubiquitous and thus invisible, no one is interested in them. Systems—tools and whole platforms even—may become obsolete because we stop caring about them in a direct way,

but this obsolescence does not make them less needed. In effect, punctualization brings with it an obsolescence of a cognitive nature: because we do not see the tool, we do not perceive it as needed, and that remains the case until something unexpected happens. Hence the OpenSSL ‘Heartbleed’ vulnerability detected in 2014, which affected the vast majority of devices connected to the Internet (Synopsys Inc. 2014). There had been an implementation bug in the wild for as long as three years until it was found in 2014 in the OpenSSL library, a key component in the encryption software that connects Linux and Windows computers to the Internet. This meant that for an extended period malicious users could have accessed encryption keys and protected user content. The software, bugs and all, was so ubiquitous that everyone had taken it for granted that it worked flawlessly. This was far from true, because there was no one to take care of the necessary code updates—the OpenSSL project did not have the financial resources for maintenance (Kaminsky 2014). The bug in the OpenSSL library that unleashed the Heartbleed vulnerability came about because of the lack of resources assigned to peer-review the code that ran the system: once it was invisible, it was only a matter of time before it became obsolete. This is also the issue in Adam’s analysis of *delegation* (2005, 2008). In exploring the origins of an ethics for things and the possibility of delegating moral decisions to so-called moral agents in an actor–network, he notes that by delegating we are giving responsibility for performing a task to either someone else, or—as here—to a digital system. Delegating a task within an actor–network can amount to an act of punctualization, because the full complexity of that task disappears from view.

Probably the lesson to learn about obsolescence is that it is not always a bad thing to implement it, and it does not necessarily conflict with the idea of environmental sustainability or even product sustainability (a.k.a. longevity). Obsolescence can also become a design strategy in its own right; for example, if price is the determining factor in the creation of an object or service, it might be convenient to use parts we know will soon be obsolete as a way to reduce the cost to the end user. It can even be a door onto a knowledge commons, for if there

is no further interest in making a profit from a technology, perhaps it could be open-sourced, so that ecologies of repair and knowledge could be built around it. All of which brings me to openness.

Openness

I have already introduced the notion of openness as a design method with various levels of application (Chapter 4). Concepts such as ‘radical openness’ imply being open not only about results, but also about the process of obtaining results and the designers’ intentions. This kind of openness—embedded in the process—is predicated on co-creation processes that engage users and designers (and eventually developers). Participation, in that sense, would be fundamental to following an open design process with an agenda that is rewritten as you go. This is what Bratton (2015) mentions when discussing the creation of ‘platform utopias’: we should ‘enable the appearance of programs that we cannot already anticipate ... in advance ... a megastructuralism based on the metaphor ... of the atmosphere and on the scale and ubiquity of the clouds’ (42). Back in the world of practicalities, I too have explored how to handle openness, in my case using licences, and how the automation of free and open licences has allowed for the development of open design as praxis. I have already presented cases that are relevant here, including Guifi.net or Debian, and of course Arduino, which is an important part of many sections in this thesis.

The historian Paul N. Edwards, who has published on the creation of technical platforms and infrastructure such as the military networks that set the ground for the Internet as we know it or the global infrastructure for weather monitoring and forecasting, takes yet another view on openness in his book *The Closed World: Computers and the Politics of Discourse in Cold War America* (1996). His interest is the development of what he calls ‘the green world’, an open space where actors try to come to terms with the world’s complexity through the ‘transcendence of rationality, authority, convention, and technology’. Edwards’ green world broadens the perspective of sustainability to cover environmental aspects. Platform designers will very rarely encounter a world that is either entirely closed or completely open (green). It is as if Edwards’ representations of the

world are just extremes that we will never reach. Will we ever, as a society, manage to achieve a green-world scenario? Have we ever experienced such a time in human history? Is the green world just a utopia, and we will always struggle to reach it from our own closed version of the world?

There is yet another aspect to openness when dealing with the idea of community: the curation of content on platforms. Who should determine how content will be displayed on the platform? Who will control the message? This leads to yet another form of closure, which Bratton calls ‘the disappearance of the outside’ that occurs when control of the message is given to those who also control the platform. In talking about Apple’s operating system strategy for mobile personal devices, Bratton describes how ‘the walled garden of iOS ... can also suffer from having to serve as both platform and content at once’. This means that, without the open curation of content, the platform itself becomes the content and is made invisible. This happens to be a bidirectional inference rule. Thus, by wanting to control the message and the way it is delivered, Apple has to be closed: ‘the price of curation is closure’ (Bratton 2015, 46). Curation warrants further discussion when dealing with communities, especially in the context of communities of learning.

Edwards’ green world (1996) is utopian, lateral, and invites exploration, and he counterpoints it with a closed world that is dystopian, vertical, and self-referential. To offer a simple explanation of what, say, a vertical world is, in the market economy we talk of segments or ‘verticals’, one of them being the Internet of things. The vertical of always-connected, always-on devices and people—the IoT—spans from the sensors doing the data-gathering all the way to the graphical user interfaces presenting information. In the closed world, it is imperative to own the whole vertical as a way to exercise control (Bratton 2015); in the open world, devices are allowed to migrate from one platform to the other thanks to open definitions of APIs, open hardware definitions of devices, or open communication protocols (Brody & Pureswaran 2014).

The study of the IoT as a platform involves a whole series of actors operating in a multiplicity of networks, which are sometimes even connected to other networks. In the IoT we find the discussion about openness happens at all levels: from the technologies involved to the data gathered, processed, and exchanged. Openness can be applied to both the immaterial and the physical existence of the IoT, but also by extension to other contexts (Herstatt & Ehls 2015, 20). The way we regulate openness is through a series of licensing schemes (Chapter 4). As Herstatt and Ehls mention in their book *Open Source Innovation*, ‘a central element in all [open] approaches is collaboration on one common valuable good ... Most open-source projects are single-user activities that probably could evolve to a flourish community’ (2015, 20). This is another relevant aspect of openness: its invitation to collaborate. Although it is just an invitation, never an obligation. Therefore, by establishing open design processes with open results, we are inviting others to join in. Intentionality is crucial; however, it will not guarantee that anyone will follow, nor that making something open will ensure a community will form around it. The question is then whether it is sustainable to have projects just because they are open, even if they lack a community to either sustain them or embrace them. In other words, does openness for the sake of openness make any sense?

Probably we can only answer that by looking at activism and politics, or perhaps by trying to construct some sort of pre-emptive code of ethics. Why would we be interested in having things open? Is it bad that ultimately a company will profit from maintaining a proprietary system that could be very beneficial for society? In light of current circumstances in data usage and how it affects users, the answer may be that we would prefer certain things not to happen. The issue with the lack of openness in technological systems such as search algorithms, control systems for self-driving cars, or encryption techniques is that it is impossible to audit them. Without a peer review of the technologies, how are we supposed to know that they are doing no harm? How can we know that they are using state-of-the-art technology and not obsolete, damaging technology? The example of OpenSSL Heartbleed shows that even when something is open, a lack of resources makes impossible to keep up with the necessary audits.

These aspects centre on the need to find strategies to make projects financially sustainable over time, a variable that I called marketability and described earlier in this chapter.

Ecology

This section deals with the concept of ecology as defined by two schools of thought: actor–network theory and cybernetics. I have chosen two of their main advocates to support my argument that the term ‘ecology’ should play a significant role in the creation of platforms. First, representing actor–network theory is the sociologist Michelle Callon, known as a leading proponent of actor–network theory and for his contributions to the field of science and technology studies, and his definition of networks; second, representing the cybernetics school, is Klaus Krippendorff, an engineer and specialist in cybernetics, language, and culture, who talks specifically about ecologies of artefacts leaving the human as an external factor. Drawing on other authors such as Ehn, I will argue that ecology is a fruitful word to apply to networks, as it brings together actor–network theory and cybernetics.

The ecology of a system is defined by the way the system is configured—the pieces, parts, people, mechanisms, locations, disposable resources, indeed anything related to the system’s existence and the way it is used define its ecology. The term builds upon Callon’s techno-economic networks (1991), which he describes as a ‘coordinated set of heterogeneous actors’ interacting to produce ‘methods for generating goods and services’ (133). Actors are social beings which relate through intermediaries. Both actors and intermediaries can be human or non-human; what distinguishes between them is the authorship role that actors assume versus the communicative role taken by intermediaries. In a techno-economic network, intermediaries are whatever defines the relationship between actors, while actors are the ones in the network that possess authorship.

The ecology of a network, as I have defined it here, would thus include all the actors and intermediaries in the network. What Callon (1991) calls a network is in a sense what I call a platform. The ecology of a platform therefore includes everything that acts as

an interface between its users, its developers, and its tools, as well as the tools themselves. It is the boundary object that augments the actor–network. As I have established, platforms are made of hardware, software, and documentation, and are tied to a community of users involved to a certain extent in the further improvement of the platform. The platform’s ecology is thus all of those things: the objects, the non-physical assets, the context, the developers, and the users.

In his 2005 book *The Semantic Turn*, Krippendorff introduces the ecology of artefacts. He describes ecology as the interaction between artefacts: how they relate to one another one-to-one or as part of a group; whether the relationship extends over time (some artefacts are the evolution of others); and how power is distributed within the relationship (in a market economy, some artefacts are parasites on others, as is the case with clones and copies). Later studies (Jung et al. 2008) add empirical data to Krippendorff’s definition. Krippendorff (2005) also explores the complexity and size of networked artefacts, and how they supersede natural ecologies on both macro and micro levels. Jung et al. concur with this understanding of ecology, noting that it ‘serves well both as a metaphor and as a theoretical construct to support the examination of complex networks of interactive artefacts’ (2008, 201). One point where all these analyses differ from actor–network theory is in considering the human factor in the ecology of artefacts. People set the terms according to which artefacts interact with one another, but are not part of the network—very different to the techno-economic, sociomaterial networks envisaged by Callon (1991), or Latour’s actor–network theory (2006). Indeed, Ehn (2007), in reviewing *The Semantic Turn*, calls for Latour’s actor–network theory’s agency in combination with Krippendorff’s ecology. I can only agree, for when considering the term ‘ecology’ we need to have humans as part of the sociotechnical assembly. Krippendorff seems to tiptoe around the human factor conversation when he says that ‘ecologies of artefacts, even of only moderate complexity, escape any one individual’s understanding’ (2005, 195), and he then moves on to possible classifications of relationships between artefacts according to different perspectives. Instead, I would argue that although I agree it is impossible to hope for a bird’s-eye view of a system because

of its size and the speed at which interactions happen (as I explained in Chapter 4), it is this that makes it imperative to consider people as part of the ecology, for our interaction with the platform is shaped by the artefacts themselves and their affordances, and vice versa. It is no more us humans controlling them, the artefacts, as it is both of us integrated in an ecology of bits and atoms with closed feedback loops (Wiener 1989 [1950]), where the evolution and duration of the platform depends on our ability as designers to participate in this always changing, strongly interconnected (Bennett 2005), and highly dependent ecology of systems and people. And a platform's ability to engage anyone other than its creators in daily interaction is manifest in its community, a term I will examine next.

Community

The term community touches upon some of the human aspects of the assemblage. Feenberg (2007) (as discussed in Chapter 2), offers a simple definition, describing the community as a world, an area of practice, 'rather than a passively observed nature to which values are ascribed' (28). Feenberg's worlds are constructed through the connections between actors, and are revealed by 'everyday experience'. From Feenberg it is a short step to Gillespie (2010), whose work I use to link communities and platforms from a political perspective, looking at how platforms should be 'progressive' and 'egalitarian'. Gillespie states that communities are governed by social contracts. These ideas—community through participation and shared experiences, progressive (democratic) and egalitarian governance, and social contracts—comprise the lens I use to view platforms in order to gauge what we should consider when studying (or forming) communities augmented by platforms. In this case, I do not stop at observations, as in Chapter 2, but dig deeper to measure those ideas against experience, to answer the question of whether the whole is greater than the sum of its parts.

Hippel (2005, 11) talks about user innovation communities, where users join together in networks 'that provide useful structures and tools for their interactions and for the distribution of innovations'. A specialist in communities of innovation—groups of people who meet to share knowledge and innovate within certain sectors—Hippel

lists the advantages that communities of innovation have over more traditional business models: speedier testing, dissemination, and the construction of larger systems through the creation of ‘interlinkable modules created by community participants’ (11). This highlights one of the reasons why modularity is relevant to platform creation, as it allows for a better way for community members to contribute, with more feasible goals than if they faced large monolithic bodies of work. The Debian case I mentioned earlier is a good example of how modularity enables a community of developers to form around solving a complex technical task (Corbet et al. 2012; Corbet & Kroah-Hartman 2017; Linux Foundation 2018). Organizing the work around a platform like Debian requires that the product itself, the software, be structured in a way that can be simultaneously modified by thousands of participants, the community. The platform is built around the idea of having groups of people simultaneously modifying text files that are strongly interconnected that a modification to one of them can affect the proper functioning of another. Such a community requires a governance model, a decision mechanism, and a series of technical tools specially created to speed up decision-making and the evaluation of contributions. Modularity is therefore at the very heart of the Debian community.

Going back to Hippel (2001, 83) and his work on innovation communities, he shows how even if in market terms it might not make sense to consider the existence of user innovation communities with the ability to innovate and compete with established market giants, the empirical data proves otherwise. Beyond innovation, another aspect that a community could tackle is dissemination in the form of knowledge-sharing. For communities of individuals this is related to openness, as described earlier in this chapter. Hippel notes that the rewards for revealing innovations within a community are ‘improved reputations, expected reciprocity, and helping to build a community’ (86). The role of the platform in this case is to enable community-sharing innovations, because in Hippel’s terms sharing will de facto build the community.

But how can a community be designed? What comes first, the community or the platform? What do designers deal with, the community per se or the process of enabling the community to happen? In his essay ‘The Future Is Not What It Used To Be’, the designer and architect Victor Papanek reflects on how the ‘function of a community ... is to act as a goal, not as a passage point; an end, not a means; a stop, not a flow’ (1988, 13). Although he is talking about communities from an urban planning perspective, there are lessons to be learnt from his essay that can be applied to platform design. When Papanek talks about ‘community design’, he describes how a successful approach is the one that puts ‘all the talent’ into creating the shared spaces, the so-called ‘communal nucleus’, and with that in place the rest of the system will follow.

One clear difference between Papanek’s empirical data (1988) and my own experience is the size of the community. The size of his ideal community—always talking about people sharing a common space—was some 500 people. Many online platforms today with so few users would not be considered sustainable. It is as if the dematerialization of human relationships in the network forces the need for a great many more users to compensate for the lack of physical substance, or, as Bratton puts it, the new utopian (city) designs are islands (closed communities) that offer ‘centralized economy of scale and density for the consumption of resources’ (2015, 41). Bratton’s ‘Stack’, a sort of global actor–network that has swallowed all the other networks and relationships, will ‘support mega-dense resource economies’, which in turn will drive the creation of larger self-contained communities. Bratton’s reference to density can be read in purely physical terms as the number of users, participants, or active contributors to the same communal resource, but it could also have a more cognitive meaning as the ratio of signal to noise, or the quality of the contributions to the community. As Geipel et al. (2014) mention in their survey of a hundred open-source communities, it is not just the number of users in a community that matters, it is the quality. Geipel et al. studied SourceForge, a repository of open-source projects, and thus had to establish their own evaluation criteria for quality. This is another

lesson to learn when working with platforms, given their ad hoc nature (they are defined by the interaction of users and developers over time): quality criteria are metrics that must be negotiated within the network. And that negotiation will have to continue for as long as the platform exists.

In addition to the ideas unpacked in this section—modularity, density, and sharing—which all have to do with community, there are two main aspects that I have dealt with in separate subsections: democracy—understood as technological empowerment—and the political—looking at governance and the political implications of platforms.

Democracy as tech empowerment

Alexander et al.’s concept of democracy (1977, 71–4) as enacted through the decentralization of power in communities, if applied to the specific situation of a socio-technological arrangement intended to build a community of co-learning in which participants educate one another by sharing experiences, participating in the discourse, and eventually contributing to the further development of the platform, makes it clear that in doing so novice users will have to be empowered by the platform’s technology in order to become active participants. This process of democratically creating technologies, expressed as a necessity by Feenberg (2010) (see Chapter 2), has been challenged both directly and indirectly by a number of designers. In this section I will suggest counterarguments to Alexander et al.’s hypothesis, and reflect on how I enacted this democratization (rather than a democratic process) in a variety of projects.

Earlier in this chapter I introduced Papanek’s communal nucleus (1988), which at first sight seems to be at risk of the classic issue of control exercised by the centre on the periphery, as explained, for example, in Law’s study (1986) of Portuguese shipping in the early modern period. However, I believe that we need to make a distinction between the technical infrastructure and the actual actor–network, the platform, which we the designers and users want to construct on top of it. This is something covered in Saldana et al.’s paper (2016), which demonstrates some of the ways to pursue a network architec-

ture, including some cases that are fully distributed. The Guifi.net case by Baig et al. (2016) illustrates how, by using the standard pieces described by Raghavan et al. (2017), it would be possible novices to participate in the construction of a network. However, in other platforms such as Arduino that have millions of users, the actual technical infrastructure is centralized with a single cloud provider. The computer architecture supporting it is distributed, but not in the way Alexander et al. (1977) suggest. I would argue that there are four possible combinations: the technical construction, distributed or not; and the platform governance, distributed or not.

It should be noted as well that the categorization of the distribution of governance cannot be defined having a binary ‘yes/no’ answer. There are different levels of how to distribute and enact power among the members of a community. This is something that Raymond registered in his book *The Cathedral and the Bazaar* (1999), which is mandatory reading for anyone interested in how an open-source software community works. In the Debian case, there is a clear hierarchy defined as the ‘benevolent dictator’ scenario (Raymond 1999, 101) where the whole community agrees to that a single person will take the final decision on whether technical patches are applied to the technology or not. Some other projects, such as Arduino, have a team of people applying the patches; they are either developers hired by the Arduino company or developers from sister communities that benefit from having a development environment (the Arduino IDE) to program their own hardware designs. Others have a fully distributed mechanism for deciding how to apply patches. Geipel et al. (2014) note the various models for command in a community of software creation: communication within the community may be centralized to a group of highly influential users and developers, and even if a high turnover in users is important for a community around a platform, it is even more important that those contributing can join in improving the tools and documentation for the newcomers that join. Figuring out which is the best model to adopt in order to achieve the platform’s development and survival is then a design decision. It is the ethos of the project that will eventually determine this aspect.

As Ehn (1988, 407) puts it, ‘The role ... skill and democracy [have] in work-oriented design is as consciously articulated values on which design should be based.’

In all these cases, there is a learning curve for community participants: no one can join one day and be elevated to benevolent dictator the next. Communities such as the ones presented here are meritocracies. Nevertheless, merit can only be obtained through active participation in the community, and that means participants have to learn how it works. For the platform’s own survival, the technical empowerment of users is essential, if only as a strategy to help novices find their feet.

There is another discussion to be continued from Chapter 2 about whether an algorithm is or is not a tool. Relevant here is the question of the democratic element in designing a platform, given that algorithms are somehow part of its creation. If software platforms are defined by algorithms, access to the processes they cover is needed for the creation, tweaking, and maintenance of the platform. There are examples where the algorithm is of vital importance for participation in a platform—Google’s search algorithm, which commands all interactions with the platform, for example. Google, the people who make searches, the people who try to boost their websites in the search rankings, the bots that try to do the same: they compose an interesting assemblage of humans and non-humans. The reason to interact with the platform is either to look for something or to try to improve one’s ranking. While the search operation is simple (it still requires certain level of skill to write the queries that will guide you to the best possible responses), the business of trying to improve your rankings calls for a series of techniques—search engine optimization (SEO)—that are not simple. In that sense, participants have no power to influence the algorithm, and they are subject to decisions made by an entity that controls the platform. The interesting point, as Rouvroy (2013) notes, is that not even the engineers running the platform know exactly how the algorithm works. So the question remains, how will democracy be enacted on platforms where algorithms or other non-humans play a vital role in how relationships are established between actors?

The political

Latour (2006, 250), talking about the possibility of change that a certain strategy may offer in the current state of (political) affairs, concludes that ‘only if forces are made of smaller ties, whose resistance can be tested one by one, ... you might have a chance to modify a given state of affairs’.⁶⁹ As designer I read this to mean that if we want our design processes, artefacts, and services to have a meaningful impact, we may as well think big but act small. This seems to be in line with Alexander et al.’s model (1977, 71–4) of distributed governance, as presented in the previous section, where democratization is explained as the process of decentralizing power to smaller units or communities. To an extent this feels as if the idea of disintermediation from Chapter 2 (Raghavan et al. 2017) is being applied to political representatives—and why should someone deal with a remote administration to solve local issues? The process of intermediation in this case seems counterintuitive. The impression is that Alexander et al.’s communities would do very well with Latour’s leaders (or engaged assemblies), practising active politics locally.

Also in *A Pattern Language*, Alexander et al. infer that nothing can be built in isolation: the action of building has to repair the world around it and within it, making the world more coherent (1977, xiii). This is a very political view of design, since it assumes there is an intention to do good, that a certain ethos is part of designing. And since design is meant to be done collectively, the ethos should be a constitutive property (or feature) of the community, and, by extension, of the platform supporting it. Linking this idea of doing good with the idea of education, and even with governance, we find Freire’s *Pedagogy of the oppressed* (2005), which suggests that the measurable goal of the education process is the ability to think-transform, and that such a goal is best pursued by reflection in action. Freire introduces his version of ‘dialogics’ as a form of education conducted by dialogue, which turns into a practice of freedom. Education, when

⁶⁹ Latour (2006): ‘if you have to fight against a force that is invisible, untraceable, ubiquitous, and total, you will be powerless and roundly defeated. It’s only if forces are made of smaller ties, whose resistance can be tested one by one, that you might have a chance to modify a given state of affairs. To put it bluntly: if there is a society, then no politics is possible.’

properly conducted, becomes a way to make everyone equal. Since technology is not neutral, the intention to educate in the creation of a platform is an attempt to do good—an attempt to practise freedom.

Other authors are concerned about the use that commercial platforms make of politics to obtain direct or indirect financial advantages. This boils down to the question of how to balance marketability and other values. In Gillespie's paper 'The politics of "platforms"' (2010) he looks at the story of YouTube and its role as enabler of freedom of speech, but also as an uncurated place where offences can be committed under the umbrella of that same freedom of speech. Gillespie shows that both positions can easily be masked by a protective layer of discourse, in which the term 'platform' plays a significant role. In much the same way, companies such as Google are working politically—through lobbyists—but also 'discursively to frame their services and technologies' (348). According to Gillespie, this is an example of how service providers use language to convey an attractive picture of their offerings, appealing in terms that go beyond pure business to impinge on other contexts, such as human rights.

Latour defines a new form of politics, separate from critical sociology, and names it the 'modern constitution'. It is 'the redefinition of politics as the progressive composition of the common world' (Latour 2006, 254), readily applied to the assemblages of society and nature at once. Latour wishes to construct the politics of actor-networks in such a way as to take both humans and non-humans into consideration. He claims that the controversies about what makes the social world should be solved by the participants, not by social scientists. Latour's political take on agency is that participants should be the ones in control of the situation.

In looking at the political aspects of communities, I have concentrated on two main topics: the idea that politics is about involvement in the governance of the community; and the idea that politics is about understanding the context when making decisions, which would lead not to the most comfortable choices, but to those that will be best for everyone (including those outside the community).

Summary

This chapter charts the lessons learnt in the course of my research, from the more theoretical to the more practical. To elucidate both the concepts and the contexts where they unfold, I sequence the appearance of the ‘functional requirements’ I would argue must be investigated when designing new platforms, and then detail the requirements themselves by function, thematizing the papers in the compilation along with other projects I have worked on.

The functional requirements are arranged in an order that displays the transition from more techno-deterministic concepts—sustainability—to those closer to collaborative action and concepts such as co-learning, collaborative work, and community. The outcome of the chapter is a series of discursive tools that platform designers can add to their personal toolboxes, to use when co-designing new platforms for sociotechnical assemblages involving humans and non-humans. The concepts constitute an actor–network of sorts, where the various terms (sustainability, obsolescence, openness, ecology, and community) are the interlinked nodes—for these concepts cannot survive in isolation. Therefore, I suggest weighing each one carefully when designing, even though, as I mentioned earlier, there is no specific order in which to apply them, and there might not be a need to use all of them in all phases of a project.

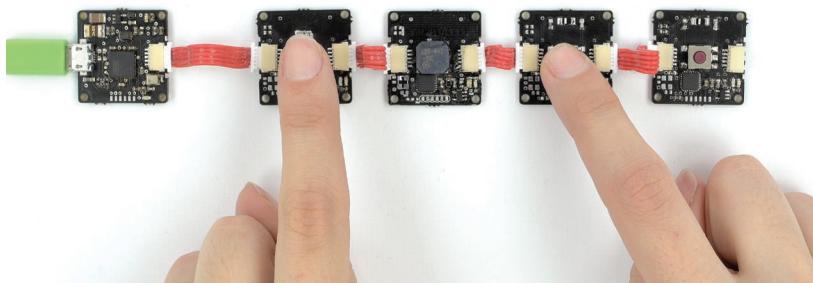


FIGURE 6
MODULAR TALKOO PROTOTYPING PLATFORM
COURTESY OF ARDUINO, 2016



6 CONCLUSIONS AND REFLECTIONS

The challenge RFID presents is how to use it to re-think human subjectivity in constructive and life-enhancing ways without capitulating to its coercive and exploitative aspects. (Hayles 2009, 48)

Hayles's point is clear: technology is a fundamental aspect of our lives, and to resist it will achieve nothing as that will merely eliminate the good as well as the bad. My concluding chapter will not attempt to solve this dichotomy; in fact, I hope I will leave readers with even more questions in forming their own understanding of what constitutes a platform and how it might be co-created as part of a democratic process.

This chapter looks at the research questions of whether modularity constitutes a basic feature for a 'platform-thing' and which is a possible set of functional requirements to create platforms. The text revisits the work done in order to answer both questions. However, this thesis is not written as a closed loop, but as steps on a ladder towards a new set of definitions to be used in real-life cases. The same steps lead to a more metaphysical discussion about actor-networks, where new types of actor are already becoming relevant almost as I write. Contemporary scholars are looking very carefully at artificial intelligence (AI), a not-so-new actor that is going to produce an imbalance in the network until we find a new equilibrium, which will require reconsideration of our own values and even of our definition of life itself. AI is one of the chief technical factors in the new platforms, as it helps make sense of the vast amounts of data they generate. The reflections in this chapter are intended to prompt the kind of open questions familiar from the international literature, and which I will continue to research in future.

Designing platforms is about building places for communities to share, co-learn, and continue to grow (where growth is an interpersonal emotional quality and not the size of the group). Their construction is collaborative in nature, and while it can emerge without the intervention of external actors, design input might be necessary to spark the process. This thesis specifically explores large-scale scenarios, where inclusive multiple prototypes were created, deployed, and tested with a variety of users. Some of the projects have achieved a certain degree of sustainability, some have evolved and been reused thanks to their transferability, and others simply vanished after a while. I have studied them over a long period of time, in response to a general understanding of digital systems that was only concerned with size, performance, and pace. As the projects evolved over time, and thanks to a methodological framework that allowed for the research aims to be written on the go (research through design), I identified a series of terms—which I term functional requirements—to be used in addition to my initial preconceptions as a way to better plan the necessary interventions to create platforms. This represents a shift from a techno-deterministic approach to a more community-centric one. It should be mentioned that I see communities as assemblages of people in an actor–network, but I do not discount the possibility of according agency to non-humans, to the point that they could play an active part of community conversations, the extent of which is discussed below.

This thesis focuses on the creation of platforms as idealized boundary objects that make actors in a network interact in optimal ways. Platforms have been seen as initially featureless; this lack of attributes is a strength, as it allows for the interactions between actors to form the platform's affordances and features. A designer's role in this scenario is one of participatory activist researcher, acting as the catalyst in the conversations among actors—whether human or not—in the network. As I suggest, designers could practise the meta-design activity of underdesigning the tools and systems in order to allow users to participate actively in the reprogramming of the platform, or even its creation in the first place. Modularity, understood as the design as

reusable blocks; density, defined as the number of users per resource, but also the signal to noise ratio; and sharing explained as part of a process of openness: all these are key aspects of platform-building.

My research question takes modularity to be central to the creation of platforms. Along the way I have discovered that not only is modularity relevant, but so are the other aspects just mentioned. Furthermore, there is a dual aspect to modularity, as it has both transferability and generalizability. Transferability refers to what Björgvinsson et al. (2012) term ‘Things’, or design outcomes that are created to be optimal in their reusability. Modularity addresses this from a broadly structural perspective. Blocks of code can be reused, protocols with clearly defined interfaces can be reused, APIs with identical method collections can be substituted for one another, and there are examples in the technical area that clearly demonstrate the extent. The technical literature addressed in the thesis points in this direction too, as when talking about constructing technical networks from standard devices, or the design of operating systems. The practical cases show also modularity to be very strong: Arduino is a modular electronics platform; the SandS modules mutated into the PELARS ones; the haptic modules were created to be mixed in different configurations made of a single design block. Generalizability, which is set on standardization, is present in the functional requirements analysed in Chapter 5. As examples, obsolescence is possible because there is a standard way to communicate on the government-regulated radio frequency spectrum (as in the case of GSM networks), and openness about the processes is a way to ensure an agreed standardization among parties competing in a certain sector. Modularity seen in terms of generalizability implies that we should be creating modules that will be standard for different platforms. Existing examples of this the authentication methods for digital platforms, payment systems for online shopping, and UI metaphors for content management systems. Given the contextualization of many of the terms used in this thesis, it is now time to revisit my research question, although first I will introduce my own definition of ‘platform’, for, following Björgvinsson et al. I will not talk of platforms, but of platform-things.

The platform-thing

Platform-thing is to platform what ‘Thing’ is to thing in Björvinsson et al.’s definition (2012). It is a boundary object that augments the communication between actors in a network, sustaining participatory decision-making by different means. As a thing, the platform should be created with transferability in mind, understood in analogy with the infrastructuring process. The platform-thing makes possible transferable processes of co-creation, including technological implementation. This concept, transferability, is easily understood in the creation of digital platforms, where there are methods—such as open licensing—that allow for the creation of transferable blocks of code or content. This can also be ported back to the physical world, as we have seen with the arrival of open licences for hardware in recent years. All of this allows for the creation of transversal platform-things that engage both humans and non-humans, as well as the physical and non-physical realms.

Since platforms are accountable for a series of values (or in my definition, functional requirements), so are platform-things. Values as such are soft properties that change over time. Platform-things should be able to accommodate different sets of functional requirements or variations of the proposed ones, as they are expected to change over time. Such variability, especially during the formation of the platform-thing, should not be a central issue, since platform-things inherit in an almost programmatic manner the properties of platforms, such as featurelessness.⁷⁰ This lack of initial definition is counteracted by a social intention. Platform-things have an embedded social API that sets them up for possible participation in shared governance, or for enrolling in the building of the platform, or creating applications on top.

My research questions revisited

In Chapter 2, I asked whether the basic property of modularity exposed by Hayles (2009) for RFID could be generalized and

⁷⁰ In object-oriented programming, we talk of different classes (definitions of variables in the system) inheriting properties from others.

transferred to actor-networks to be used to build platforms, now platform-things. I approached this concept with a view to reprogramming the actor-network by creating a new set of rules to get the various modules to interact in such a way as the result would be greater than the sum of its parts. This is what should constitute a platform. My first research question is thus whether this modularity, which we might refer to as transferability (in the sense that its affordances could be transferred to other designs) and generalizability (its properties could become a standard), constitutes a basic feature for a ‘platform-thing’—a sociotechnological construct for the running of reconfigurable applications—yet at the same time a sociomaterial artefact for collaborative learning?

Given my definition of platform-thing as a transferable outcome for a design activity in the more specific context of digital platforms, modularity is thus a basic property, because it is in the nature of code to be modular. It then falls to the designers and developers to under-design the libraries and basic foundations of the platform—its blocks of code—to keep them sufficiently modular to be transferable. In contemporary software design practice we talk about libraries, APIs, snippets, all of which signify different forms of the transferability of code and data between developers, but also between entities.

On the other hand, generalizability often depends on whether a platform is widely adopted, something that is related to its sustainability, where the size of the community ensures it has the necessary financial resources to keep the system running. It can also depend on the ability of platform designers to negotiate with other emerging platforms to join forces in creating a single way to do that one action. An example taken from the technical world is the creation of a special interest group (SIG) to study a communication protocol and manage the IP related to its implementation. Generalizability can thus be achieved either through direct growth or through the general growth of the network that the platform is part of.

Thus I can answer my main research question in the affirmative: platforms, and more specifically platform-things, have the key cha-

racteristics of reusability and reprogramming, which makes them modular by design, and also transferable and generalizable.

Regarding the second research question, ‘Given the definition of a “Thing” by Björgvinsson et al. (2012) as a “socio-material assembly”—and my own definition of platform–thing—which kind of functional requirements can lead the design work towards the creation of a platform?’ I have responded to the question by looking at a new set of terms (sustainability, obsolescence, openness, ecology, and community), which are now part of the new designer’s toolbox for the co-creation of platform–things.

Visions of possible futures

While strong AI is still far from realized (Boden 2017), which would imply the artificial reproduction of a consciousness with human qualities, the notion of an assemblage’s collective intelligence is still evident in the concepts framed by Bennett (2005), Rovroy (2013), Bratton (2015), and even Latour (2006), as if there was a superior meta-being capable of controlling the direction of the platform’s centre of gravity, despite the efforts of any of the actors to move in a different direction. This vision suggests that we should look at platforms as living entities: as always changing co-learning spaces, sociotechnical actor–networks with agency as a whole but also as parts, systems that are somehow self-managed through the interaction of all of the actors with one another. Thus, we can then also start to look at how the roles in a platform could be understood in entirely new ways, challenging traditional conceptions of control (Wiener 1989 [1950]), ownership (Lessig 2001, 318), and even the very nature of life (Conde Pueyo 2014, ch. 1). The evolution of systems towards this reality implies that each one of the actors involved is in constant transformation. In what follows I will highlight some of those transformations and how they might affect the future of platforms.

People as sensors

One of the most visible aspects of this transformation of the roles of actors involved in a platform is how people are no longer mere

beneficiaries of the output of systems, but are increasingly involved in the generation of data for the platforms, a movement that started with the transformation of media consumers into media creators (Baigorri et al. 2005, 165), and that has now evolved with the shift in importance from content to meta-data (Rouvroy 2013). As long as we talk about the creation of anthropocentric design, platforms will be there to serve human needs—which have to be expressed in machine terms and given to the platform for processing. This automatically makes humans into sensors (Afolabi et al. 2017), the extensions of the machine who capture information from the world and give it to their non-human partners for them to process.

There are sets of transducers that humans use to capture information from the world. Personal communication devices are now pervasive, and by the inclusion of sensor technology and the process of capturing all kinds of contextual information, they have made us humans into virtual sensors, or extensions of the non-human entity we call a machine (Resch 2013). We have transformed our own role from one of mere users or consumers of results from the processing operations, to producers of the information needed to feed those operations. The platform acts as an intermediary between us and ourselves, while at the same time it depends on us to feed it, for its own survival. And while the technical platform lacks the strong AI to reach a certain consciousness and self-awareness (Boden 2017), the assemblage does have hundreds of humans helping to build the collective consciousness of the platform.

While we have the choice not to subscribe to the idea of acting as a sensor, we already do it for the sake of society. Most of us will agree to give data to medical research, or publish images of our lives online, or approve of the automated detection of speeding vehicles on public roads; images and data that are then scrutinized by an algorithm to extract patterns that will help the evolution of automatic delegation machines (Adam 2005). To my mind, the challenge is not whether we will accept this new role of humans in an assemblage, but whether we will be able to negotiate acceptable conditions for most of us. This is a complex issue that transcends the purely cognitive aspect described here, as it affects our ability to decide, the social reality of class (those

with the power to affect and those who do not), the creation of new types of labour and labour laws, and so on.

The currency of belonging

There are two aspects to the currency of belonging: the purely financial, and how much are we willing to surrender our privacy for the platform (Leckner 2018). From a financial perspective, things cost money: bandwidth, servers, the workforce to keep the technical aspects of the platform running, and the like. From this point of view, belonging to a platform will mean someone will have to pay the bills. If the user will not, who will? And this the other aspect, if users are not paying money, could they be paying in another manner? Would those payment conditions be acceptable?

Currently, platforms are analogous to banks, being intermediaries between the material and the immaterial realms. Humans represent savers, but also borrowers' intent on consumption. Money being saved is the data, while money being lent is the information. The nature of money changes the moment it has a purpose; intentions are the meta-data of money, what contextualizes it. Meta-data is what makes the data information: the location where a picture was taken, the browser that accessed a website, the gender of the person who completed a purchase. For the platform, the meta-data is almost more important than the data itself, since it helps build relations between users, thus defining patterns and assigning value (Rouvroy 2013; Brody & Pureswaran 2014; Mantelero 2016). Contemporary platforms earn money from the aggregation of the immaterial and use it to pay for the material. They put their trust in us users to build up enough information through our interaction with their systems for them to have something to sell afterwards. The value is not so much in the unique value of a few scraps of information about us, but in the interaction of that unique value with the rest from other individuals (Mantelero 2016). Our real individual value is very low (Brody & Pureswaran 2014). More than ingots of gold, our information represents the copper coins in the bank cashier's drawer.

But value is not bidirectional, as the potential pay for what we give away for belonging is really low. We can get very little for our infor-

mation, at least for as long as people keep on trying to produce as much data as possible, posting their lives on platforms, sending transactions to databases, geolocating their every movement. Even if some of the data strikes us as nonsensical or disconnected, the platform will potentially make sense of it (Simon 1996, 8). On the other hand, the time we have to invest to become untraceable (while not sacrificing the use of digital technology) or to erase our traces, represents quite a lot of work that translates into a real expense for us. Platforms are not free, and sharing in their benefits comes at a price. The economy of platforms, especially when they are large, is based on hope and huge capital investments. These investments come from financial funds that look for a return on their investment. Sometimes they accept data as part of the transaction—when it comes to data we are just the coins in someone's pocket, and this will become increasingly apparent as time passes (Arrieta Ibarra et al. 2018).

Software

The importance of software in the process of building platforms is crucial. The mere choice of a programming language determines the capabilities and even the affordances of the user interface. The success of a platform depends in part on its design, something not covered in the 'Lessons learnt' section of the thesis, as it is not an aspect to be taken into account on a conceptual level, but rather on a cognitive level. It is not a given, but it is strongly dependent on aesthetic values and technical performance at a certain point of time. I touched upon the idea of software when dealing with terms such as obsolescence and marketability. Contemporary digital platforms rely heavily on AI, and need new software paradigms, larger technical infrastructures, databases of unimaginable size, and data transfers larger than life. The software needed to command such technical infrastructure, search data, or trace relation graphs between relevant categories is very complex. Since data is not always available in the same way and the machine is not concerned with events (Rouvroy 2013), the answers to my questions may not be deterministic. Software has to act as an interface to help us separate the relevant from the irrelevant, so its design is crucial, and fields such as data visualization are already of uttermost importance in every scientific field.

The software for any new platform thus has to satisfy the needs of many: the simple user interface for the end consumer/human sensor; access to constantly flowing data streams in the search of information (events); the analysis of patterns and decision-making by AI engines; the visualization of information for those controlling the validity of the algorithms; the ease of development and the cost of the tools for those creating the platforms. Future software for platform-making is one of the biggest design challenges we are currently facing. Large corporations are already offering institutions and individuals the chance to build using their proprietary software tools on top of their own big data infrastructures, but that is still far from optimal. There are already indications that software will be one of the most active fields of work for interaction designers in the near future.

Protocols

Protocols describe the communication process between the different actors in the assemblage: humans with humans, non-humans with non-humans, but also between humans and non-humans. Protocols even help establish intra-platform communication mechanisms. Protocols give the reference for how communication will be established, which patterns of information have to be exchanged for a valid exchange to take place, and the formats for encapsulating data. By establishing standardized ways to exchange data, we are also opening for the possibility of letting platforms grow, since they could, say, syndicate knowledge bases in order to improve their ability to realize certain tasks. Standardized communication mechanisms could also help create ways to audit a platform's ethical aspects. I can imagine the creation of ethics APIs, used to exchange information with regulatory bodies in an automated fashion to make sure operations follow the law. The more we can delegate these platform actions, the more we can focus in improving other qualities, or in actively participating in the co-learning of new aspects that might be relevant to platforms.

State machines

Among the non-humans participating in platforms, in the future we are going to find more and more digitally enhanced artefacts. Technology can be embedded in everyday devices, adding new kinds of functional possibilities we have not thought of. But the possibilities

of digital technology are also to provide devices with their own reactive behaviours to variables from the environment. State machines are control mechanisms that allow assigning behaviours to simple machines without having to orchestrate their responses from a central authority at all times. For example, a lift should be able to move up and down in a building without having to obtain approval from a central control unit.

Small devices typically do not run an operating system—in other words, their software is not general purpose in nature, but rather a simple executable that can be changed by reprogramming. That reprogramming is done with a communication protocol imprinted in the device when it is manufactured. For example, an Arduino board protocols is part of the so-called bootloader, a piece of software whose only function is to reprogram the device.

Simple human delegations can be translated into state machines that reproduce a scripted series of commands multiple times. But given that we humans already act as sensing devices for platforms, could not we be seen as executing simple scripts especially programmed for us? Have platform developers worked out how to hack into human consciousness by implanting state machines in our clothing and getting us to perform involuntary tasks to feed the platform with data? Leaving aside the conspiracy theories, could it be possible that the collective consciousness of the platform has made us into our very own state machines, designed in such a way as we can morally accept their existence?

Algorithmic sovereignty

In contemporary digital society, control is enacted through machines commanded by dynamic programmatic structures. These structures, which we call algorithms (Dourish 2016), operate using data that has been deprived of meaning (Rouvroy 2013), and therefore has become irrelevant to us whilst it makes us insignificant, as our singularity is diluted in matrixes of aggregated raw information flows (Mantelero 2016), automatically categorized by the algorithm itself. Rouvroy presents the difference between judgement and critique when talking about AI. Judgement describes the ability to cluster data into catego-

ries. Critique, on the other hand, defines the possibility of challenging the categories to better accommodate the data. Both actions, up to the arrival of AI, were done by us humans, but that has changed now. AI is a system that never stops, where error does not exist, where events are washed away, and where categories are defined automatically by DL algorithms according to parameters we can never understand. This makes impossible for us to challenge the categories, which precludes us criticizing anything, including the machine itself. In this case, neutrality excludes not just one type of human, but the human race per se. This is a new type of neutrality where we are all equal to the machine, but not equal with the machine—the concept to challenge through algorithmic sovereignty.

While hardware is complex in nature, to the point that it is not yet auto-generative, software and the abstraction that it derives from (meaning algorithms) can be produced by software. The level of delegation we have in software is unlike almost anything else before in history in scale and responsibility. However, regulatory efforts will be put in place at some point, since this is how our socioeconomic system works (Dutton 2014). Once there is a potential point of conflict, society builds a new failsafe switch to try to control the situation. There are historical examples that range from the creation of laws to stop unjust cases to the evolution of political systems that are more humane and up-to-date with our understanding of the world. It is only a matter of time until new standards and regulations start policing our new AI-centric reality. But if that is going to happen, where then is the risk?

The production of data is embodied. We are the sensors when we are one with our pervasive measuring tools: some comes from our daily interactions with others; some of it is generated by our self-quantification devices; some of it is a trace left when conducting financial transactions with the always-traceable, always-ubiquitous digital money. It is worth asking ourselves if it makes sense to work for a future where humans work in data farms, feeding self-thinking machines in a sort of symbiotic relationship. It is hard to understand how a symbiosis with algorithms could work. It is much easier to imagine a parasitic relationship, where algorithms extract the value

from us by forcing us to abandon what is relevant to focus on making what they need. The mechanism we choose in order to keep control is what I like to call algorithmic sovereignty, or the ability to exercise power through algorithms, the main abstraction within the human-machine assemblage our society is becoming.

Platform design in a time of AI

This thesis starts from the premise that platforms emerge where user needs meet the developers' ability to deliver systems accommodating those needs. Furthermore, I have shown that platforms are boundary objects between stakeholders in actor-networks. In a sense, platforms use the networks' meta-data to create new digital contexts, as participation in platforms implies closing a behavioural loop between the system and the user both as an individual and a part of a collective, from personal and aggregated data alike. Platform-things are designed to be modular, generalizable and transferable, acting out scenarios if use that, once abstracted in the form of patterns, can be reused. AI can make predictions and analyse possible scenarios using so-called brute force combinatory strategies, weighed against one another to decide the best possible outcome based on the metrics used to train the expert system. Could not then a trained AI algorithm create platform-things based on existing designs or patterns? While my research thus far has demonstrated the inherent complexities when designing platforms, given the computational ability to anticipate (and therefore simulate) the behaviour of systems, how much of the work of the creation of platform-things could be taken over by algorithmic intelligence? The real question, having analysed the state of the art of AI technology, is no longer whether artificial systems will be able to synthesize new platforms, given an existing set of design patterns to enable the interaction between humans, for such a possibility seems plausible. The real question is whether AI will be able to anticipate new modalities of interaction, and alternative user journeys for communicative purposes, in what in essence would constitute new artificial designs for platform-things.

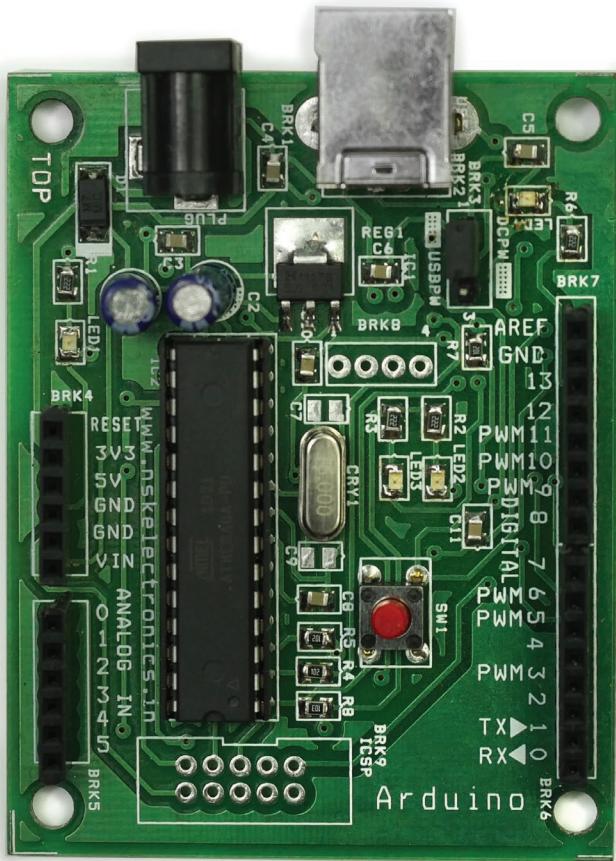


FIGURE 7
ARDUINO COMPATIBLE BOARD FOUND
AT A STREET MARKET IN BANGALORE, INDIA,
2011

REFERENCES

- Ackermann, J. R. (2009). Toward Open Source Hardware. *University of Dayton Law Review*, 34(2), 183–223.
- Adam, A. (2005). Delegating and distributing morality: Can we inscribe privacy protection in a machine? *Ethics and Information Technology*, 7(4), 233–42, doi:10.1007/s10676-006-0013-3.
- (2008). Ethics for things. *Ethics and Information Technology*, 10(2–3), 149–54, doi:10.1007/s10676-008-9169-3.
- Aesthetics and Computation Group (2003). *Design By Numbers*. <http://dbn.media.mit.edu/> [25 Feb. 2017].
- Afolabi, O., K. Driggs-Campbell, R. Dong, M.J. Kochenderfer, & S.S. Sastry (2017). *People as Sensors: Imputing Maps from Human Actions*. ArXiv ID: 1711.01022.
- Akrich, M. (1992). The de-scription of technical objects. *Shaping Technology Building Society. Studies in Sociotechnical Change*, 205-224. Cambridge/London: The MIT Press.
- Alexander, C., S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King & S. Angel (1977). *A Pattern Language*. New York: OUP.
- Andersson Schwarz, J. (2017). Platform Logic: An Interdisciplinary Approach to the Platform-Based Economy. *Policy and Internet*, 9(4), 374–94, doi:10.1002/poi3.159.
- Andreessen, M. (2007). *The three kinds of platforms you meet on the Internet*. http://pmarchive.com/three_kinds_of_platforms_you_meet_on_the_internet.html [1 Mar. 2017].
- Arduino Forum (2011). *vinciDuino—un Arduino Leonardo hecho por gente del foro*. <http://forum.arduino.cc/index.php?topic=78781.0> [4 June 2017].
- (2012a). Moved From *Re: vinciDuino—un Arduino Leonardo hecho por gente del foro*. <http://forum.arduino.cc/index.php?topic=92869.0> [4 June 2017].
- (2012b). *vinciDuino RevC up and running—Leonardo clone*. <http://forum.arduino.cc/index.php?topic=89769.0;nowap> [4 June 2017].
- (2017). *Arduino Forum—Index*. <http://forum.arduino.cc/> [23 July 2017].
- Arrieta Ibarra, I., L. Goff, D. Jimenez Hernandez, J. Lanier & E. G. Weyl (2018). Should We Treat Data as Labor? Moving beyond "Free". *AEA*

- Papers and Proceedings*, 108, 38-42, doi: 10.1257/pandp.20181003.
- Associated Press (2018). *Facebook's Zuckerberg says regulation of social media firms is 'inevitable'* | CBC News. CBC. <https://www.cbc.ca/news/technology/facebook-zuckerberg-users-privacy-data-mining-house-hearings-1.4614174> [24 July 2018].
- Ayass, M. & J. Serrano (2012). The CERN Open Hardware Licence. *International Free & Open Source Software Law Review*, 4(1), 71–8, doi:10.5033/ifosslr.v4i1.65.
- Baig, R., L. Dalmau, R. Roca, L. Navarro, F. Freitag & A. Sathiaseelan (2016). Making Community Networks economically sustainable, the guifi.net experience. In GAIA '16: *Proceedings of the 2016 workshop on Global Access to the Internet for All*, 31–6, doi: 2940157.2940163. Florianopolis, Brazil: ACM.
- Baigorri, L., D. Bravo, K. Brunet, M. Cañada, D. Casacuberta, K. Cascone, J. Cortell, J. de la Cueva, J. L. de Vicente, R. Díaz López, A. Estalella, F. Felipe, F. G. Gil, P. Jiménez, P. D. Miller, A. Orihuela, P. Sanz Almoguera, F. Tolstoi & D. Villar Onrubia (2005). *Creación e inteligencia colectiva*. Sevilla: Zemos98.
- Barragan, H. (2004). *Wiring: Prototyping Physical Interaction Design*. Ivrea: I.D.I.I.
- Bateson, G. (1972). *Steps to an ecology of mind*. Northvale, New Jersey, London: Jason Aronson Inc.
- Bennett, J. (2005). The agency of assemblages and the North American black-out. *Public Culture*, 17(3), 445–66, doi:10.1215/08992363-17-3-445.
- Björgvinsson, E., P. Ehn & P.-A. Hillgren (2012). Design Things and Design Thinking: Contemporary Participatory Design Challenges. *Design Issues*, 28(3), 101–116, doi:10.1162/DESI_a_00165.
- Blikstein, P. & A. Sipitakiat (2011). QWERTY and the art of designing microcontrollers for children. *Proceedings of the 10th International Conference on Interaction Design and Children—IDC '11*, 234–237. Place: Publisher doi:10.1145/1999030.1999070.
- Boden, M. A. (2017). *Inteligencia Artificial*, i: October. Madrid: Turner.
- Bratton, B. H. (2014). Black Stack. *E-Flux* (53).
- (2015). Cloud Megastructures and Platform Utopias. In J. Geiger (ed.), *Entr'acte*, 35–51. Place: Palgrave MacMillan.
- Brody, P. & V. Pureswaran (2014). Device democracy: Saving the future of the Internet of Things. *IBM Global Business Services Executive Report*.
- Brown, L. (2017). Standard YouTube License vs. Creative Commons. <https://filmora.wondershare.com/youtube-video-editing/standard-youtube-license-vs-cc.html> [28 May 2017].
- Callon, M. (1991). Techno-economic networks and irreversibility. In J. Law

- (ed.), *A sociology of monsters: Essays on power, technology and domination*, 132–165, London: Routledge.
- Candeira, J. (2006). Barrapunto | Entrevista a David Cuartielles, de Arduino. barrapunto.com. <http://barrapunto.com/articles/06/09/28/1541241.shtml> [4 Nov. 2017].
- Casas, R., D. Cuartielles, J. L. Falcó & L. Malmborg (2002). Positioning technologies in learning. In *Proceedings—IEEE International Workshop on Wireless and Mobile Technologies in Education*, WMTE 2002. 161–162.
- — — — — & Á. Marco (2004). Location Based Services for CSCL: Micromobility, positioning paper. In *Workshop on Spatial Awareness in Collaboration and Group Interaction at CSCL SIG*. EPFL Laussane, Switzerland.
- — — — — & H. J. Gracia (2007). Hidden issues in deploying an indoor location system. *IEEE Pervasive Computing*, 6(2), 62–9.
- Chatterton, P., D. Fuller & P. Routledge (2007). Relating action to activism: Theoretical and methodological reflections. In S. Kindon, R. Pain & Mike Kesby (eds.), *Participatory action research approaches and methods: Connecting people, participation and place*, 216–22. London, New York: Routledge.
- Conde Pueyo, N. (2014). *Biological Computation in Yeast*. Barcelona: Universitat Pompeu Fabra.
- Conniry, K. L. (2016). *National Security, Mass Surveillance, and Citizen Rights under Conditions of Protracted Warfare*, doi: 10.15760/etd.3195. Portland: Portland State University.
- Corbet, J., G. Kroah-Hartman & A. McPherson (2012). *Linux Kernel Development: How Fast it is Going, Who is Doing It, What They are Doing, and Who is Sponsoring It*. Linux Foundation white paper, August. San Francisco: The Linux Foundation.
- — (2017). *Linux Kernel Development Report 2017*. San Francisco: The Linux Foundation.
- Cox, J. (2017). *A new robot-powered ETF AEIQ is beating the stock market*. <https://www.cnbc.com/2017/10/20/a-new-robot-powered-etf-aeiq-is-beating-the-stock-market.html> [20 Feb. 2018].
- Creative Commons (2017). *Share your work—Creative Commons*. <https://creativecommons.org/share-your-work/> [28 May 2017].
- Creswell, J. W. & V. L. Plano Clark (2011). *Designing and Conducting Mixed Methods Research*. Los Angeles: SAGE.
- Cuartielles, D., L. Malmborg & P. Schlaucher (2003). High Tech Hunters and K3 Nomads. In *HCI International 2003*. Crete, Greece.
- (2004). Resign desearch: The Darwinian evolution of contemporary thought species. In P. Ehn & J. Lowgren (eds.), *Design [x] research: Essays*

- on interaction design as knowledge construction.* Malmö: Malmö University Press.
- & L. Malmborg (2004). Sound-based augmentation of spaces. In *Workshop on Spatial Awareness in Collaboration and Group Interaction at CSCL SIG*. EPFL, Lausanne, Switzerland.
- — & P. Schlaucher (2004). Collaborative Sketchpad: Students designing educational technologies. In *IADAT 2004—First Conference on Innovation, Technology and Research in Education*. Bilbao, Spain.
- (2010). *Arduino Intro for kids at FARO—Medea*. <http://medea.mah.se/2010/06/arduino-intro-for-kids-at-faro/> [25 Feb. 2017].
- & X. Yang (2010). *Oh_Oh Experiments With Robots—Medea*. http://medea.mah.se/2010/08/oh_oh-experiments-with-robots/ [25 Feb. 2017].
- (2011). *The power of the copy of the copy—Medea*. Medea. <http://medea.mah.se/2011/04/the-power-of-the-copy-of-the-copy/> [2018-07-24].
- N. de la Riva Irieipa, I. Gallego, X. Yang & E. Gallego (2011). *Arduino Robot*. <https://www.arduino.cc/en/Main/Robot> [4 Jan. 2017].
- A. Göransson, T. Olsson & D. Sjunnesson (2011). *1scale1 SweetBlue Wiki*. <https://github.com/1scale1/sweetbt/wiki>.
- (2012). *You may all play music, but you are not a band—on collaboration, skill learning, and our education system—Medea*. <http://medea.mah.se/2012/05/you-may-all-play-music-you-are-not-a-band/> [Feb. 2017 25].
- A. Göransson, T. Olsson & S. Stenslie (2012a). Mobile haptic technology development through artistic exploration. *Haptic & Audio Interaction Design*, 31–40. Lund, Sweden: Springer-Verlag.
- — — — (2012b). Developing Visual Editors for High-Resolution Haptic Patterns. *The Seventh International Workshop on Haptic and Audio Interaction Design*, 42–4. Lund, Sweden: HaptiMap.
- & D. Taylor (2013a). *Underverk: SmartWatch, a hacked IDE to program the Sony SmartWatch 1*. <https://github.com/underverk/SmartWatch> [4 Jan. 2017].
- — (2013b). Delivery number D2.1 *Datasheets for SandS Motherboard and Modules*. Malmö: Social&Smart.
- A. Göransson, T. Olsson & S. Stenslie (2013c). Telehaptic Awareness. In *Proceedings of the 7th conference on Tangible, embodied and embedded Interaction*, TEI 2013. 1–8. Barcelona, Spain: ACM.
- (2014a). *Delivery number D2.2. Report on Thinking Appliance Manual*. Malmö: Social&Smart.
- (2014b). How Deep Is Your Love? On Open-Source Hardware. In P. Ehn, E. M. Nilsson & R. Topgaard (eds.), *Making futures: Marginal notes on innovation, design and democracy*, 153–70. Cambridge, MA: MIT Press.

- & A. Göransson (2015). *Professional Android wearables*. Indianapolis, IN: Wrox.
- E. Katterfeldt, G. Dabisias & A. Berner (2015). *Delivery number 4.2: Report on Final STEM Learning Kit with Integrated Learning Analytics for Trials*. Malmö: PELARS.
- D. Nepelski & V. Van Roy (2018). Arduino: A global network for digital innovation. In *Open Innovation 2.0 Yearbook 2017–2018*, 15–24. Luxembourg: Publications Office of the European Union.
- Cubitt, S. (2014). Telecommunication Networks: Economy, Ecology, Rule. *Theory, Culture & Society*, 31(7–8), 185–99, doi:10.1177/0263276413511490.
- Dourish, P. (2016). Algorithms and their others: Algorithmic culture in context. *Big Data & Society*, 3(2), 205395171666512, doi:10.1177/2053951716665128.
- Dunne, T. (1999). *Hertzian tales: Electronic products, aesthetic experience and critical design*. London: Place: Publisher.
- & W. Gaver (2001). *The Presence Project*. London: CRD Research.
- Dutton, W. H. (2014). Putting things to work: Social and policy challenges for the Internet of things. *Info*, 16(3), 1–21, doi:10.1108/info-09-2013-0047.
- Edwards, P. N. (1996). *The Closed World: Computers and the Politics of Discourse in Cold War America*. Cambridge, MA: MIT Press.
- Ehn, P. (1988). *Work-Oriented Design of Computer Artefacts*. Stockholm: Arbetslivscentrum.
- (2007). The semantic turn: A new foundation for design. *Artefact*, 1(1), 56–9, doi:10.1080/17493460600844157.
- Feenberg, A. (1991). Questioning Technology. *Whole Earth Review*, (73).
- (2001). *Questioning Technology*. London: Routledge.
- (2007). Between Reason and Experience. *Danish Yearbook of Philosophy*, 42, 7–32.
- (2010). Democratic Rationalization: Technology, Power, and Freedom. In id., *Between Reason and Experience: Essays in Technology and Modernity*. Cambridge, MA: MIT Press.
- Fischer, G. (2003). Meta-Design: Beyond User-Centered and Participatory Design Department of Computer Science, 430 UCB 2 Design Time and Use Time 3 User-Centered Design and Participatory Design. *Human Computer Interaction: Theory & Practice (Part I)*, 88–92.
- Freire, P. (2005). *Pedagogy of the oppressed*. New York: Continuum.
- Fry, B. & C. Reas (2007). *Processing: A Programming Handbook for Visual Designers and Artists*. Cambridge, MA: MIT Press.
- Furlong, E. E., K. J. Boose & S. T. Boysen (2008). Raking it in: the impact of

- enculturation on chimpanzee tool use. *Animal Cognition*, 11(1), 83–97, doi:<https://doi.org/10.1007/s10071-007-0091-6>.
- Gammon, N. (2013). *How to use this forum—please read*. <https://forum.arduino.cc/index.php?topic=148850.0> [15 May 2017].
- Gaver, B., T. Dunne & E. Pacenti (1999). Design: Cultural probes. *interactions*.
- Gaver, W. (1991). Technology affordances. In *Proceedings of the SIGCHI conference on Human factors in computing systems Reaching through technology—CHI '91*. 79–84.
- A. Boucher, S. Pennington & B. Walker (2004). Cultural probes and the value of uncertainty. *Interactions* 11(5), 53–6, doi:<http://dx.doi.org/10.1145/1015530.1015555>.
- (2012). What Should We Expect From Research Through Design? In *CHI '12 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 937–46. Austin, Texas: ACM.
- Geipel, M. M., K. Press & F. Schweitzer (2014). Communication in Innovation Communities: An Analysis of 100 Open Source Software Projects. *Advances in Complex Systems*, 1725(8), doi:<10.1142/S021952591550006X>.
- Gibson, J. J. (1986). The Ecological Approach to Visual Perception. *Journal of the Society of Architectural Historians*. New York: Psychology Press.
- Gillespie, T. (2010). The politics of ‘platforms’. *New Media & Society*, 12(3), 347–64, doi:<10.1177/1461444809342738>.
- Givens, A. D. (2013). The NSA Surveillance Controversy: How the Ratchet Effect Can Impact Anti-Terrorism Laws. *Harvard National Security Journal*.
- Goodman, D. J. & R. A. Myers (2007). *Analysis of Patents Declared as Essential to GSM as of June 6, 2007*. Tomball, TX: Fairfield Resources International, Inc.
- Google Developers (2011). *Google I/O 2011: Android Open Accessory API and Development Kit (ADK)—YouTube*. <https://www.youtube.com/watch?v=s7szcpXf2rE> [2017-03-6].
- Gran, A., J. Holst & M. Ochsmann (2005). *ARDUINO meets PROCESSING*. http://web.archive.org/web/2014052222104/http://webzone.k3.mah.se/projects/arduino-workshop/projects/arduino_meets_processing/instructions/ [25 Feb. 2017].
- Greengard, S. (2015). *The Internet of Things*. Cambridge, MA: MIT Press.
- Göransson, A. & D. Cuartielles (2012). *Professional Android Open Accessory Programming with Arduino*. New York: Wiley.
- Hafner, K., H. C. Ritter, T. M. Schwair, S. Wallstab, M. Deppermann, J. Gessner, S. Koesters, W.-D. Moeller & G. Sandweg (1991). Design and Test of an Integrated Cryptochip. *IEEE Design & Test of Computers*, 8(4), 6–17,

- doi:10.1109/54.107201.
- Hale, C. R. (2001). What is Activist Research? *SSRC*, 2(2), 13–15.
- Hayles, N. K. (2009). RFID: Human Agency and Meaning in Information-Intensive Environments. *Theory, Culture & Society*, 26(2–3), 47–72, doi:10.1177/0263276409103107.
- Helmond, A. (2015). The Platformization of the Web: Making Web Data Platform Ready. *Social Media & Society*, 1(2), doi:10.1177/2056305115603080.
- Herstatt, C. & D. Ehls (2015). Open Source Innovation: *The Phenomenon, Participant's Behaviour, Business Implications*. New York: Routledge.
- Hippel, E. von (2001). Innovation by User Communities: Learning from Open-Source Software. *MIT Sloan Management Review*, 42(4), 82–6.
- (2005). *Democratizing Innovation*. Cambridge, MA: MIT Press.
- Hixon, T. (2013). For Most Small Companies Patents Are Just About Worthless. *Forbes*. <https://www.forbes.com/sites/toddhixon/2013/10/04/for-most-small-companies-patents-are-just-about-worthless/#424ab0e53ef3> [26 May 2017].
- Hooker, B. & S. Kitchen (2004). *Edge Town*. <http://www.benhooker.com/edgetown/> [25 Feb. 2017].
- Huhtamo, E. (2010). Thinkering with Media: On the Art of Paul DeMarinis. In P. DeMarinis (ed.), *Buried in Noise*, 33–9. Heidelberg: Kehrer.
- Johnson, D. G. & K. W. Miller (2008). Un-making artificial moral agents. *Ethics & Information Technology*, 10(2–3), 123–33, doi:10.1007/s10676-008-9174-6.
- Jung, H., E. Stolterman, W. Ryan, T. Thompson & M. M. Siegel (2008). Toward a framework for ecologies of artefacts: How are digital artefacts interconnected within a personal life? *Proceedings of the 5th Nordic conference on Human-Computer Interaction: Building bridges*, 201–210. Place: Publisher. doi:10.1145/1463160.1463182.
- Kaminsky, D. (2014). *Be Still My Breaking Heart | Dan Kaminsky's Blog*. <https://dankaminsky.com/2014/04/10/heartbleed/> [18 Nov. 2017].
- Kaptelinin, V. & B. Nardi (2012). Affordances in HCI: Toward a mediated action perspective. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, 967–76. Austin, TX: ACM.
- Katterfeldt, E. S., M. Cukurova, D. Spikol & D. Cuartielles (2018). Physical computing with plug-and-play toolkits: Key recommendations for collaborative learning implementations. *International Journal of Child-Computer Interaction*, doi:10.1016/j.ijcci.2018.03.002.
- Kobayashi, S. (2006). *GAINER.cc | Main/Home*. <http://gainer.cc/> [25 Feb. 2017].
- T. Endo, K. Harada & S. Oishi (2006). GAINER: A Reconfigurable

- I/O Module and Software Libraries for Education. In *Proceedings of the 2006 International Conference on New Interfaces for Musical Expression* (NIME-06), 346–51. Paris: Nime.
- Krippendorff, K. (2005). *The semantic turn: A new foundation for design*. Boca Raton: CRC Press.
- Krueger, B. S. (2005). Government surveillance and political participation on the Internet. *Social Science Computer Review*, 23(4), 439–52, doi:10.1177/0894439305278871.
- Lane, M. (2014). *Greek and Roman Political Ideas*. London: Penguin.
- Latour, B. (1996). On actor–network theory. *Soziale Welt*, 47(4), 369–82.
- (2006). Reassembling the Social: *An Introduction to Actor-Network Theory*. New York: Oxford University Press.
- Law, J. (1986). On the Methods of Long Distance Control: Vessels, Navigation, and the Portuguese Route to India. *Power, Action and Belief: A New Sociology of Knowledge? Sociological Review Monograph* (32), 234–63.
- (1992). Notes on the Theory of the Actor–Network: Ordering, Strategy and Heterogeneity. *Systems Practice*, 5(1992), 379–93.
- & Mol, A. (2001). Situating Technoscience: An Inquiry into Spatialities. *Society & Space*, (19), 609–21.
- Leckner, S. (2018). Vem är positiv till insamling av användargenererad data på internet? In U. Andersson, A. Carlander, E. Lindgren & M. Oskarsson (eds.), *Sprickor i fasaden*, 55–70. Gothenburg: SOM-institutets.
- Leigh Star, S. & J. R. Griesemer (1989). Institutional Ecology, ‘Translations’ and Boundary Objects: Amateurs and Professionals in Berkeley’s Museum of Vertebrate Zoology, 1907–39. *Social Studies of Science*, 19(3), 387–420, doi:<https://doi.org/10.1177/030631289019003001>.
- Lessig, L. (2001). The future of ideas: *The Fate of the Commons in a Connected World*. New York: Random House.
- (2004). *Free Culture: How big media uses technology and the law to lock down culture and control creativity*. New York: Penguin.
- Levinson, J., J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling & S. Thrun (2011). Towards fully autonomous driving: Systems and algorithms. *IEEE Intelligent Vehicles Symposium, Proceedings*, 163–8, doi:10.1109/IVS.2011.5940562.
- Linux Foundation (2018). *The Linux Foundation*. <https://www.linuxfoundation.org/> [28 Apr. 2018].
- London, B. (1932). *Ending the Depression through planned obsolescence*. New York: B. London.
- Lury, C. & N. Wakeford (2012). *Inventive Methods*. Abingdon: Routledge.
- Löwgren, J. & B. Reimer (2013). *Collaborative Media: Production, Consumption*.

- tion, and Design Interventions*. Cambridge, MA: MIT Press.
- Mack, C. A. (2011). Fifty Years of Moore's Law. *IEEE Transactions on Semiconductor Manufacturing*, 24(2), 202–207, doi:10.1109/TSM.2010.2096437.
- Maeda, J. (1999). *Design by Numbers*. Cambridge, MA: MIT Press.
- Malinen, T., T. Mikkonen, V. Tienvieri & T. Vadén (2010). Open source hardware through volunteer community: A case study of eCars—now! In *Proceedings of the 14th International Academic MindTrek Conference on Envisioning Future Media Environments—MindTrek '10*, 65. Tampere, Finland: ACM.
- Mantelero, A. (2016). Personal data for decisional purposes in the age of analytics: From an individual to a collective dimension of data protection. *Computer Law & Security Review*, 32(2), 238–55, doi:10.1016/j.clsr.2016.01.014.
- Marcuse, H. (1969). Repressive Tolerance. *A Critique of Pure Tolerance* (1965), 95–137, doi:10.1080/02691721003749901.
- Mari, E. (1974). *Autoprogettazione?* Milano: Enzo Mari.
- (2017). *Autoproyectos?* 2nd edn, Santiago de Chile: Editorial Hueders.
- Mattern, F. (2005). Ubiquitous Computing: Scenarios for an informatized world. *Fleisch*, 1–13.
- Meissner, J. L., A. Strohmayer, P. Wright & G. Fitzpatrick (2018). A Schnittmuster for Crafting Context-Sensitive Toolkits. *Proceedings of the 2018 CHI Conference*, (April), doi:10.1145/3173574.3173725.
- Mellis, D., S. Jacoby, L. Buechley, H. Perner-Wilson & J. Qi (2013). Microcontrollers as material. *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction—TEI '13*, 83–90, doi:10.1145/2460625.2460638.
- Mogensen, P. (1992). Towards a Provotyping Approach in Systems Development. *Scandinavian Journal of Information Systems*, 3(1), 31–53.
- Mr. Robot (2016) ‘h1dden-pr0cess’, season 2 episode 10 [television series].
- Muntadas, A., O. Rofes, M. A. Staniszewski, J. Arnaldo, J. Lebrero Stals, R. Lozano-Hemmer, S. Ramsay, C. Giannetti, D. Walla, A. Mercader, V. Roma, K. Neray, A. Böröcz, M. Verhulst, A. Frank, J. I. Roca, M. Jaukkuri, C. Scoates, B. Weil, S. Lamuniere, J. Angel, B. Mari, P. Landry, A. Efimova, A. Urlus, A. Fresneda, J. P. Fajardo, S. Wirtzsch, C. Ruello, C. Cannizzaro, N. Germain, G. Fustenberg, C. Maestrali, H. D. Christ, I. Dressler & C. Borelli (2002). *Muntadas: On Translation*. Barcelona: ACTAR.
- Nelson, G. (1967). Obsolescence. *Perspecta*, 11(1967), 170–6.
- Norman, D. A. (1990). *The Psychology of Everyday Things*. New York: Basic Books.
- Olsson, T., A. Göransson, S. Stenslie, D. Cuartielles & D. Sjunnesson (2012).

- Technological Mashups: Building HiFi wearables. *IMAC 2011 Proceedings*, 16–20.
- Open Source Hardware Association (2012). *Open-Source Hardware FAQ—Open Source Hardware Association*. <https://www.oshwa.org/faq/> [4 Nov. 2017].
- Osterman, K. F. (1998). *Using Constructivism and Reflective Practice To Bridge the Theory/Practice Gap*.
- Papanek, V. (1988). The Future Is Not What It Used to Be. *Design Issues*, 5(1), 4–17.
- (2011). *Design for the Real World: Human Ecology and Social Change*. 2nd edn, London: Thames & Hudson.
- Pellizzoni, L. (2015). *Ontological Politics in a Disposable World: The New Mastery of Nature*. Surrey: Ashgate.
- Plantin, J.-C., C. Lagoze, P. N. Edwards & C. Sandvig (2016). Infrastructure studies meet platform studies in the age of Google and Facebook. *New Media & Society*, doi:10.1177/1461444816661553.
- Powell, A. (2012). Democratizing production through open source knowledge: From open software to open hardware. *Media, Culture & Society*, 34(6), 691–708, doi:10.1177/0163443712449497.
- Raby, F. (2000). *Project #26765 Flirt*. London: RCA CRD Research.
- Raghavan, B. & S. Hasan (2012). Macroscopically Sustainable Networking: An Internet Quine.
- Raghavan, B. & D. Pargman (2017). Means and Ends in Human–Computer Interaction: Sustainability through Disintermediation. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 786–796. Denver, CO: ACM. doi:10.1145/3025453.3025542.
- Raymond, E. S. (1999). *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. 2001 revis. Sebastopol, CA: O'Reilly.
- Redström, J. (2001). Designing everyday computational things. PhD diss., Gothenburg studies in Informatics 20 (May).
- Resch, B. (2013). People as Sensors and Collective Sensing: Contextual Observations Complementing Geo-Sensor Network Measurements. In J. M. Kriss (ed.), *Progress in Location-Based Services*, 391–406. Berlin: Springer.
- Resnick, M., F. Martin, R. Berg, R. Borovoy, V. Colella, K. Kramer & B. Silverman (1998). Digital manipulatives: New toys to think with. *Communications of the ACM* (April), 281–7, doi:10.1145/274644.274684.
- Rettig, M. (1994). Prototyping for tiny fingers. *Communications of the ACM*, 37, 21–27, doi:10.1145/175276.175288.
- Rittel, H. W. J. & M. M. Webber (1973). Dilemmas in a General Theory of Planning. *Policy Sciences*, 4(2) (June), 155.

- Rouvroy, A. (2013). Society of the Query #2—Antoinette Rouvroy: Algorithmic Governmentality and the End(s) of Critique. Place: Publisher.
- Rudd, J., K. Stern & S. Isensee (1996). Low vs high-fidelity prototyping debate. *Interactions*, 3(1), 76–85, doi:10.1145/223500.223514.
- Saldana, J., A. Arcia-Moret, B. Braem, E. Pietrosemoli, A. Sathiasselan & M. Zennaro (2016). *Alternative Network Deployments: Taxonomy, Characterization, Technologies, and Architectures*. Place: Publisher.
- Schuppli, S. (2014). *Deadly Algorithms | Radical Philosophy*. <https://www.radicalphilosophy.com/commentary/deadly-algorithms> [1 Mar. 2017].
- Schön, D. (1983a). From Technical Rationality to Reflection-in-Action. In *The Reflective Practitioner: How Professionals Think in Action*, 21–69. Place: Publisher.
- (1983b). The Structure of Reflection-in-Action. In *The Reflective Practitioner: How Professionals Think in Action*, 128–67. USA: Basic Books, Inc.
- (2001). The Crisis of Professional Knowledge and the Pursuit of an Epistemology of Practice. In D. Raven & J. Stephenson (eds.), *Competence in the Learning Society*, 185–207. New York: Peter Lang.
- Seravalli, A. (2014). While Waiting for the Third Industrial Revolution: Attempts at Commoning Production. In P. Ehn, E. Nilsson, & R. Topgaard (eds.), *Making futures: Marginal notes on innovation, design and democracy*. Cambridge, MA: MIT Press.
- Shiffman, D. (2009). Interview with Casey Reas and Ben Fry | Rhizome. Rhizome Blog. <http://rhizome.org/editorial/2009/sep/23/interview-with-casey-reas-and-ben-fry/> [7 Oct. 2017].
- Simon, H. A. (1996). *The sciences of the artificial*. 3rd edn, Cambridge, MA: MIT Press.
- Spikol, D., N. Ehrenberg, D. Cuartielles & J. Zbick (2015). Design strategies for developing a visual platform for physical computing with mobile tools for project documentation and reflection. In *CEUR Workshop Proceedings*, 57–62. EU: CEUR-WS.org.
- Stallman, R. (2002). Free software, free society: *Selected essays of Richard M. Stallman*. Boston: GNU Press.
- Stenslie, S., A. Göransson, T. Olsson & D. Cuartielles (2014). Stitchies—Towards telehaptic performativity. In *TEI 2014—8th International Conference on Tangible, Embedded and Embodied Interaction, Proceedings*, 327–9. Munich, Germany: ACM.
- Stephenson, N. (1999). *In the Beginning... Was the Command Line*. New York: Avon Books.
- Sterling, B. (2005). *Shaping Things*. Cambridge, MA: MIT Press.
- Stout, D. & T. Chaminade (2009). Making Tools and Making Sense: Complex, Intentional Behaviour in Human Evolution. *Cambridge*

- Archaeological Journal*, 19(1), 85–96, doi:<https://doi.org/10.1017/S0959774309000055>.
- Stringer, E. T. (2014). *Action research*. 4th edn, Thousand Oaks, CA: SAGE.
- Sun, S.-L. & G. A. Barnett (1994). The international telephone network and democratization. *Journal of the American Society for Information Science*, 45(6), 411–21, doi:10.1002/(SICI)1097-4571(199407)45:6<411::AID-ASI7>3.0.CO;2-J.
- Synopsys Inc. (2014). *Heartbleed Bug*. <http://heartbleed.com/> [18 Nov. 2017].
- Säljö, R. (2010). Digital tools and challenges to institutional traditions of learning: Technologies, social memory and the performative nature of learning. *Journal of Computer Assisted Learning*, 26(1), 53–64, doi:10.1111/j.1365-2729.2009.00341.x.
- Thompson, N. (2018). *Mark Zuckerberg Q&A: The Facebook CEO Talks Cambridge Analytica, the Company's Problems, and Big Data* | *Wired*. <https://www.wired.com/story/mark-zuckerberg-talks-to-wired-about-facebooks-privacy-problem/> [24 July 2018].
- USAID (2016). Technical Note: *The 5rs framework in the program cycle*. Place: Publisher.
- Weiser, M. (1991). *The Computer for the 21st Century*. *Scientific American*, 265(3), 94–104, doi:10.1038/scientificamerican0991-94.
- Wiener, N. (1989). *The Human Use of Human Being*. IEEE Transactions on Information Theory; London: Free Association Books (first pub. 1950).
- Yahoo Help (2017). *Change your photo's license in Flickr* | *Yahoo Help—SLN25525*. <https://help.yahoo.com/kb/SLN25525.html> [28 May 2017].
- Yeomans, G. (2014). *Autonomous vehicles—Handing over control: Opportunities and risks for insurance*. London: Lloyd's.
- Zacchiroli, S. (2011). Debian: 18 years of free software, do-ocracy, and democracy. In *OSDOC '11 Proceedings of the 2011 Workshop on Open Source and Design of Communication*, 87–7. Lisbon, Portugal: ACM.
- Zimmermann, Y. (1998). *Del Diseño*. 2nd edn, Barcelona: Gustavo Gili.

Dissertation Series:

New Media, Public Spheres and Forms of
Expression

Culture and Society

School of Arts and Communication, K3

Malmö University



Fig 8 Mural by Boamistura, Zaragoza, Spain.
Courtesy of Diego Cuartielles



978-91-7104-942-1 (print)

978-91-7104-943-8 (pdf)

MALMÖ UNIVERSITY
205 06 MALMÖ, SWEDEN
WWW.MAU.SE

DAVID CUARTIELLES

PLATFORM DESIGN APPENDIX

Creating Meaningful Toolboxes When People Meet

TEKHNOLOGIA OMNIPOTENS REGNAT

•XIII•



MALMÖ
UNIVERSITY

PLATFORM DESIGN
APPENDIX

Dissertation Series: New Media, Public Spheres and Forms of Expression

APPENDIX

Doctoral dissertation in Interaction Design

Faculty: Culture and Society

Department: School of Arts and Communication, K3

Malmö University

Information about time and place of public defense, and electronic
version of the dissertation: <http://hdl.handle.net/2043/26130>

CC BY-NC-SA David Cuartielles Ruiz, 2018

Copy editor (chapters 1-6): Charlotte Merton

Illustration*: Julia García

Cover, bleed design and layout: Laura Balboa

*Adapted from original artwork by Boamistura

ISBN 978-91-7104-942-1 (print)

ISSN 978-91-7104-943-8 (pdf)

Printed by Holmbergs, Malmö 2018

This work is licensed under the Creative Commons Attribution-
NonCommercial-ShareAlike 4.0 International License. To view a copy
of this license, visit <http://creativecommons.org/licenses/by-nc-sa/4.0/>
or send a letter to Creative Commons, PO Box 1866, Mountain View,
CA 94042, USA.

DAVID CUARTIELLES

PLATFORM DESIGN APPENDIX

Creating Meaningful Toolboxes When People Meet

Malmö University, 2018



CONTENTS

APPENDIX A, PUBLICATION 1	7
APPENDIX B, PUBLICATION 1	27
APPENDIX C, PUBLICATION 1	39
APPENDIX C, PUBLICATION 2	191
APPENDIX C, PUBLICATION 3	235
APPENDIX D, PUBLICATION 1	279
APPENDIX D, PUBLICATION 2	293
APPENDIX D, PUBLICATION 3	299



APPENDIX A, PUBLICATION 1

Appendix A, Publication 1

D. Cuartielles, Resign desearch: The Darwinian evolution of contemporary thought species. In P. Ehn & J. Lowgren (eds.), *Design [x] research: Essays on interaction design as knowledge construction* (Malmö: Malmö University Press, 2004).

RESIGN DESEARCH: THE DARWINIAN EVOLUTION OF CONTEMPORARY THOUGHT SPECIES

David Cuartielles

School of Arts and Communication, Malmö University

david.cuartielles@k3.mah.se

ABSTRACT

From its origin as a craft, passing through the market reappropriation of the term during the 80's, to the moment when many disciplines have adopted it as a generic creative strategy, design has taken many forms and has incurred a series of ideological transformations. In an attempt of making sense within the already established structures in the fields of science and academic practices, some authors suggest the creation of the area of design research through the methodology of systematic inquiry.

This text (first) analyzes the evolution of design as presented by different design practitioners, design philosophers, and design theorists. After studying the etymological definitions for both Design and Research according to two contemporary scholars, I will depict my understanding of the contemporary academic design scene through a historical overview, thus taking an evolutionary approach to the concept of Design Research.

The text ultimately concludes by counterattacking the position of systematic inquiry applied to design research by starting from the original statement of design: to provide with solutions, making use of the argument of the western-centered background of the scientific knowledge, and presenting cases that I have faced in my everyday design practice as part of a design collective.

With Resign Desearch I try to address that Gestalt is a big part of design, a part that contains an ideological discourse that is as hard to leave out of design practice as it is to find it in scientific knowledge. This contextualization politicizes design research to the point of making it partial. As a matter of fact, it isn't until the postmodern era that design found a way of coping with the market. Therefore I believe that there is room for creating a research discipline with a different character—more contextualized—than any of the other scientific disciplines.

The interesting evolutionary characteristic of design resides in the fact that the different forms it has taken since its origins are coexistent nowadays. The use of the term »Darwinian evolution« in the subtitle to this paper is therefore intentionally ironic. It tries to address that despite all the controversy around the different kinds of work within design (commercial, research, educational, social, etc.) there is room for all of them.

SEMILOGY OF DESIGN

I have mainly worked with two authors in looking for the origins of the word design: Yves Zimmermann, a practitioner who has published collections of essays about design methodology in Spanish, and Vilém Flusser, a media philosopher whose professional development occurred mostly in Brazil.

Both Zimmermann (1999) and Flusser (1999) open up their argumentation by deciphering the etymological coding within the term design. In English design is both a noun and a verb; as a noun it means »intention«, »plan«, »aim«, »basic structure«, which corresponds to the Spanish word »designio« as Zimmermann mentions. All those terms are connected to cunning and deception. As a verb, meanings include »to stimulate«, »to draft«, »to sketch«, etc. The word is derived from the Latin »signum«, meaning »sign.«

There is a clear difference in the way the discussion goes from there. For Flusser, an exiled Jew who ran away from Prague to London at the age of twenty, design appears as a link between disciplines. He compares Greek and German terms for bringing design, art, and technology together. Terms like the Greek »mechos«, a device designed to deceive, connect to the German »Macht« (»power«, »might«) and »mögen« (»to desire«, »may«). Thus a machine is a device conceived to deceive. »Techne«, which is the Greek root for technology, means »art.« He says that according to Plato, »artists and technicians were tricksters, because they seduced people producing distorted versions of ideas.« »Ars« is the Latin equivalent to »techne.« Since »ars« means the ability to turn something to one's advantage, »artifex« (artist) would mean trickster. At the same time the German term for art, »Kunst«, de-

rives from »können«, the modal verb meaning »to be able« which makes the artist the one able of doing something.

In Flusser's mind »design«, »machine«, »technology«, »art«, and »ars« coexist and have a common existential view of the world. He claims that since the Renaissance there has been a tendency of separating those terms into different disciplines, which provoked a gap between the scientific and the aesthetic knowledge production. This conceptual paradigm used design to cover the gap between art and technology. This is why design has the strength to become the basis of all culture, to deceive nature by means of technology. In Flusser's idealized vision of design, the designer is the only one with the ability of showing the tricks and deceptions constructed by art and technology.

Zimmermann, a Swiss designer who has been practicing in Spain for more than forty years, makes an extensive study of the etymology of the term »design« referring to all the main European languages: Spanish, French, German, Italian, English, and even Latin. In Spanish the two words »designio« and »diseño« have evolved from the original one, the Latin »signum.« The prefix »de« is originally Latin and means »to belong to«, »to be in reference to.« In the context of design, the designation of something is the selection of the signs that characterize the object's identity. In »diseño« we find a different prefix, »di«, which in contrast to the previous one comes from Greek and here indicates the thing that has a meaning, a signature.

From French we get again two terms, »dessin«, the word design itself, but with the more generalized meaning of drawing. The second one is »dessein«, the equivalent to the Spanish »designio.«

Germany has had a very strong influence in design. As a matter of fact, in many English manuscripts we find the word »Gestaltung« as a generic expression for design. This is what the English noun design means explicitly. The term's root »Gestalt« refers to »the generic aspect« of things, »the profile« of something, what makes it be what it is, and nothing else. Clearly this first meaning is closer to the idea of uniqueness from »signature« or »signum«, in Latin, parallel to yet again another word in German, »Absicht«, or »aim«, »intention.« Here the author emphasizes its root, »Sicht«, which in German means »the view of something«, and thus relates the concept to its visual form.

Italian offers expressions that are very close to the original Latin ones. »Designare« is translated as »to assign«, or »to draw«; this means that the object gains its signature through the action of drawing. On the other hand we have »disegnare«, the verb to the noun »Disegno«, the design.

Unlike other languages, English presents a single term that means both the activity of design production and the design product itself. Therefore it is hard to translate the richness of meanings arising from that single word.

DEFINITIONS

Design

Design, according to Zimmermann, is defined as a profession practiced by graphic designers, architects and industrial designers. It is through success stories that the other disciplines have adopted the term in order to get the medial status that designers got at a certain moment in history. The traditional practice of design included the configuration of 2D and 3D objects, especially the ones produced industrially.

Ken Friedman, a design theorist studying the ways of creating a common ground for developing design research, is not in disagreement with this idea, but he positions himself at a different level of discussion, since he is not interested in the ideological discourse of design as much as in the possibilities for generating a research discipline out of it (Friedman, 2003).

For him, design is an activity that can be abstracted and not only a profession. Design refers to a goal-oriented process, aiming to solve problems, which have different dimensions. On the one hand, design is understood as a profession, with its discipline characterization, and its areas of inquiry. On the other hand, design presents a theoretical dimension, arising from its interdisciplinary, integrative nature. With the terms »integrative discipline«, Friedman means that design is embedded as part of many disciplines. Nowadays we find designers in fields that go from engineering to linguistics. At each one of those disciplines design is a field of practice and applied research, and when abstracted from each discipline it turns into a field of thinking and pure research.

Many authors like Flusser, Zimmermann, or Chaves (who will be mentioned later), manifest a clear ideological agenda without which design cannot be conceived. Among these, Zimmermann is the one who most clearly shows his aim by saying that: »The object's signature should determine its use and usability.« He states that the main purpose for a design project should be the use of the design result, and that »every object should pass a truth-test about its use.« In one of his essays he even refers to Wittgenstein: »the use is the truth.« I conclude with his reference to the object's configuration as the signature composed by the addition of all the possible uses of it. If the object is designed according to its »designio,« then

it is usable and therefore it is a true object. An object with total usability would be »perfect.«

Research

Friedman uses a dictionary definition of the word »research«, thus presenting it as the collection of methods that allow us to use the tools constituted by theory. As applied to our case, it allows us to abstract and conceptualize within design. I find particularly interesting his way of disjoining the word research into the prefix »re« and the core »search.« »Re« is not an English term, as he correctly writes, it is a prefix indicating repetition. Taken from Latin, the prefix »re« is part of many verbs in contemporary Spanish and means to iterate the action of the verb that follows. Examples are: »apropiar« means to appropriate, »reapropiar« to appropriate again. He adopts this analysis to make a distinction between basic research and applied research, the first one involving a search for general principles, while the later adapting the findings of basic research to classes of problems. Finally he introduces clinical research that takes the results from the other two fields and applies them to specific situations. In design terms, clinical research generates design cases, or examples of application of research results, and this case generation is the kind of daily work of design practitioners.

For the pragmatic Zimmerman, research within design happens as a side effect of the designer's need to know the reality surrounding the design itself. Designers look for solutions to problems, which implies the need to know about the details that characterize those problems. It seems for this author that the designer's duty is to produce satisfactory solutions, which collides with one part of the scientific method when, e.g., working with hypothesis testing. The solution of hypothesis testing is in the level of truth of a logical statement. Therefore, a statement could be either true or false, which means that, in case of it being false, the solution to the problem would not be found, and therefore denying the design as such. When iterating within the hypothesis testing process it should be possible to find some solution, or to do some kind of simplification of the problem.

The design result can only take form within or through a material, yet Zimmerman proposes the example of even using language as a material for a design project. Even if it is not mentioned in his text, it appears that he is implicitly accepting the possibility of working with materials that do not have a physical manifestation. This opens the floor to design practices, services, and others, and goes beyond mere objects.

To summarize, the design theorist Friedman and the design philosopher Zimmermann have contrasting opinions on the role of research in design. The later gives importance to it as a tool for getting a proper understanding of the problem's context. Friedman, instead, believes that there are links between design cases coming from very different disciplines that allow the creation of a generic theory of design.

Design Research

Having shown the origins and etymological definitions for both Design and Research as presented by two contemporary scholars, I now choose to analyze their relationship through a historical overview, thus taking an evolutionary approach to the concept of Design Research, itself the main topic of this paper. In the description of the term research, I made a brief reference to what Friedman and Zimmermann consider the relationship between design and research to be. Both visions are in opposition and I don't fully agree with any of those.

Since Design Research is a field in its definition process, I believe that there is still a chance to shape it in order not to let it get trapped into another academic spider-web. As a design researcher I can agree with Friedman that we need to have mechanisms for collecting, analyzing, and representing data in the right way. But it is Friedman's expressed understanding of »the right way« what I dislike. He has a very engineered view on what design should be, almost retro-modernistic utopian view of design as the world's superhero. About the value of artistic research he writes: »I believe that a study of design based on profound knowledge embraces the empirical world of people and problems in a deeper way than purely self-generated artistry can do.« When saying this he is disqualifying all the origin of design, which is strongly based in the relationship between art and technology, as I will explain in my historical overview of the discipline under study. At the same time he shows a total lack of sensibility towards a different research method, the one based in the introspection and self-reflection of the solo artist.

By the end of the text I will introduce a design research methodology trying to link Zimmermann's pragmatist ideas about research as informer, with Friedman's knowledge generation methods.

EVOLUTIONAL HISTORY OF THE IDEOLOGY OF DESIGN

The ideology of design has transformed since its origins. The reason for analyzing design from this specific point of view is that I have a strong belief in the fact that design should always rest on a series of ideological principles. Ideology is strongly linked to a subjective understanding of systems. One might ask then about how to make design research in the way Friedman defines it, as an objective scientific activity of gathering data. Since design research is such a young area within design, I think that the historical revision of the discipline's ideology can help to answer this question. Ideology, when understood in the generic terms that the author I refer to is using, becomes more or less like the code of ethics of doctors, architects, or any other professions. It ends up being a list of good intentions constrained by the other factors in play.

It is for this reason that I have decided to look into the writings of another design practitioner: Norberto Chaves. I have chosen to work with practitioners' visions of the design profession because design has grown from a craft. It therefore seems that the political values of the work of those practitioners should be taken in consideration when trying to define the field of design research. All the three practitioners I have chosen have witnessed design's growth since the late-modernism. All of them have been conditioned by their surroundings and were forced to reflect on the ethical values of their profession at least twice: in the shift from modernism to market centered design, and in the one from the later to postmodernism.

Chaves' main concern is the contradiction that exists within the social consciousness in design practice under certain socio-economical conditions. He makes a strong critique towards contemporary design, where »people seem to produce without questioning, trapped by mere language games or trends« (Chaves, 2001).

In his historical revision on the origin of design as such, it appears as a way of questioning existing ideological structures, and also the techniques and processes of cultural production. Design was born as the culture within industry carrying an aim of social transformation. From the beginning it was proposing a new way of producing objects, and a new way of production of the industrial objects. Between that period and contemporary design we find many differences. For instance, we have learned to understand that form does not necessarily follow function. The only thing in common is that there are still design objects resulting from design processes. The consciousness and the processes of production are totally different.

Within this, we perceive ideology as a social discourse, a stream of collective consciousness generated by certain material conditions that mediate the behaviors

and ideas of the social groups or the society in general. Ideology is explained as a necessary characteristic with the final goal of describing and making feasible the relationships that justify certain social practice or the general social order as a whole. Its function is not to look for the details of the socio-economical structures, but to generalize its particular features. It works as the legitimate interpretation of the relationship between design and society.

The ideology of design pioneers

The modernists constructed their ideological model on top of the rational, humanist, universalistic, utopian, idealist, moralist, mechanistic, elitist, avantgardistic ... discourse that was dominating the general line of thought of the time.

It was the cultural elite who challenged the existing cultural paradigm and proposed a revision of the creative project. They proposed a cultural revolution adjusting the world of the symbolic to the technical and social reality. It started within architecture and the products for the habitat (understood as the living environment) and quickly spread to the rest of the material production. Architects created the perfect habitat for that user not considering his economical conditions and real politics of housing production. Designers were practicing a utopia. And we can find many examples especially in modernistic architecture and its sub-product, the infamous socialistic architecture.

At that time in history, design was mainly practiced within pre-industrial activities and its ideology would take one or more of the following manifestations:

- functionalistic discourse: or the relationship user-object
- technical discourse: relationship product-process
- economical discourse: relationship product-cost
- abstractionist discourse: relationship form-sense

In essence this movement was mainly an ideological one. It was lacking a theoretical apparatus that could put the idea of design under a scope of critical analysis. Their main supposition was their main failure too: to think that there are objective values in the objects that can give them a meaning by themselves. This is obviously the idea of Gestalt. In a way this was showing certain ingenuity where the designers would imagine a generic idealized »User« that was never involved directly in the design process. The designer would develop his/her work according to the imagined objective needs of that user model. That model was characterizing a physical and physiological entity lacking an own history and socially formed cultural preferences. It was not coincidental with any specific segment of the population and was of course following the basic principles of modernity.

The ideology of the market

With the advent of the market society, design was no longer something hard to integrate in society, something exotic and idealized. It became a basic tool of its time through the market economy. There were design producers, distributors, and consumers, all of whom could be measured in terms of economical variables, and they had a value. The agency for the development of design resided within the market itself, and those supporting it: industries, corporations and the organisms regulating the markets. There has been a re-appropriation of the term design and a totally new ideology has been attached to it, having nothing in common with the original discourse. If we could consider the utopian modernistic design as the one developing an »ingenuous reasoning«, the ideology of the market would be the one practicing »pragmatic reasoning.« Economic savings within the production process would be the reason to introduce design and design products by many manufacturers. Concerning the discussion around the ideological aspects of design vs. the economic ones, Chaves says that »no business man had to read Le-Corbusier for incorporating design dynamically and actively into industry.«

This is an entirely new understanding of society forcing us to rewrite most of the terms that had composed the basic ideology of the design pioneers:

- society: market
- user: consumer
- design quality: aggregated value
- design object: product
- product: merchandise
- design proposal: offer
- use needs satisfaction: buying motivation
- rationality: competition

We would then start to consider rational the objects, products or services reaching the market that is our society's rationality. At the same time, the rationality of production is not to produce something that is useful or providing us with the ideal service, but to produce something that can be consumed in both senses of the word: consume as use, and consume as exhaust until we need to get a new one. That will bring a design in harmony with the market and therefore be rational.

Under these circumstances the designer's role is to innovate, not any longer to cover the user's needs, but to provide the market with a new attractive event.

The post-avant-garde discourse

Contemporary design is strongly influenced by the post-modern line of thought that places the rationalist efficiency under judgment. This new current was born in the late 70's and is not so different from the first ideological model, with the difference of having an alibi for using the market as an ally.

Postmodernism acts as a sort of over-design in fields where the market economy acts slowly or is even paralyzed due to its inability to introduce radical innovations.

The real postmodern aspect of the contemporary situation is that there is no longer an emotive identity between the product and the producer. It is our consciousness the one creating that distance, since there is no longer anything that lasts forever. This taking distance is described almost as a kind of objectification. It could be done in such a way that we could recommend, as designers, not to make a design, since the existing objects are enough to cover the needs. And this is, in my opinion, the ultimate proof of being consequent when designing, being able of recognizing that someone else did it better and earlier as we envisioned it.

The way of acting socially within design is going through entities that play outside the market. They come through other entities, interest fields, or organizational possibilities of the population, allowing the designer to work under different conditions.

As a practitioner in the contemporary context, the postmodern designer cannot invent solutions without counting with the existing actors that have direct access to the problems. If one wants to go social, it is mandatory to look for ways to be in contact with those having a design need. At the same time, social design consists in taking whatever design assignment is presented and making it as good as possible trying to illustrate for the client the relevant aspects that such a piece will have for society. It is not enough to decline jobs that are not social.

After the death of post-modern design

Chaves' exploration of the design's history ends with post-modernism. But obviously the history of design continues. I'd like to think that the line that follows the historical chapter is the one drawn by the user-centered tradition. Norberto Chaves introduced his reflections upon design in a conference in Buenos Aires in 1988, precisely the same year when Pelle Ehn presented his doctoral dissertation. While Chaves' views upon design were quite generic but inspired by the practice of graphic design, Ehn's work is centered in artifacts with the computer science tradition. The main characteristic of this Scandinavian design line is going to be how to include

the user as part of the design process. This approach was created with a clear ideological aim.

Human-centered design, user-centered design, participatory design, and other modalities of contemporary design practices are based in the idea of democracy in design, where users become part of the process of generation of ideas for new designs. What is interesting is that all those methods appear mainly in the computer science departments at the universities, but they are slowly being absorbed by the rest of the academic community, as part of the computerization of the educational centers during the 80's and 90's. As a matter of fact, I wouldn't dare to say that this trend of human related design implies a general trend in design as strong as postmodernism or any of the other schools of thought, yet. But I consider it important since I think that the different tendencies in design are nothing but a different way of reading social aspects from our everyday reality, and that as such they can coexist as people with different opinions can inhabit the same space: with moments of tension, and moments of peace.

During the 90's the RCA in London took a new approach to design research. The work of Gaver, Dunne and Raby, Hooker and Kitchen, among others shows a new sensibility within design. To be more specific they started to work with electronic devices (artifacts) as their main focus. Following their publications one can draw a line sketching the evolution in their way of thinking. They have defined, or at least brought to a bigger audience, the ideas of critical design, and conceptual products. They conceive design as a way for opening discussion and not only as a way of introducing new usable products in the market. In the *Presence* project (Gaver et al., 2001) they introduced new ideas on methods for gathering information from users, cultural probes, tools that opened again a discussion about if design research should have its own way of doing research. In *Design Noir* the authors present a model for contemporary ideology in design (Dunne and Raby, 2001). The production process should be informed by values based on ways of understanding the world or reality. Design could be understood as either affirmative design, experimental design, or as critical design. Obviously affirmative design is the one that reinforces how things are now. Experimental design is somewhere in between and tries to extend the medium through the novelty of the concepts. Critical design is not necessarily interested in the industrial production, nor exploring the novelty of the aesthetic qualities, but would try to explore social, psychological, cultural, technical and economic values.

Here they make an interesting comment pointing out that architecture has maybe been the field that was already working with these issues. This seems to collide

with the vision that the other authors presented earlier would have about architecture, considered as an immobile field, and therefore open for experimental design.

The main problem to be found in this kind of post-post-modern design ideology is that it does not address the market directly and does not have a clear and easy industrialization process. It will never be truly popular. At the same time it is not enough to just offer an alternative, it is necessary to make things accessible to industry and challenge its technological agenda. Therefore critical designers should not just offer design proposals, but feasible product ideas that could inform the consumers about certain issues.

The authors define a category of design products that expresses their point of view. They call it »Design Noir« and it would focus on how electronic products could expand the psychological dimensions of experiences. They talk about conceptual products vs. conceptual design. The difference is in the existence of the design itself, a product has to exist and should be delivered to a user, while a conceptual design could just take the form of a sketch or a text. It is the user experience that the designer looks for, and the object should provoke existential moments.

For this to happen designers have to change their focus from the aesthetics of production to the aesthetics of consumption, something that I interpret as the need of introducing an understanding of the use/consume psychology in the design process. It is through working with those issues that it is possible to utilize design as a tool for social critic.

A conceptual product brings a narrative attached to it. It is a fictional piece that shortens the distance between the user and the product. Of course the designer will need to reach the suspension of disbelief, forcing the user to wonder about how much is true in the product, reflection that will hopefully make the user think about the nature of the design.

Anyway, these products are kind of art pieces that are not intended to reach a big market; Dunne defines them as »products for the mind [that] provide mental pleasure and stimulate reflection.« At the same time, he tries not to be totally unrealistic with who would within the design market have the chance to work this way. He proposes »academic« designers as the main source of conceptual design products, since they can »exploit their privileged position to explore a subversive role for design as social critic.«

At the same time, in this last book and in their website titled *EdgeTown* (Hoek er and Kitchen, 2003) they play with the idea of collections of pieces that contribute to the suspension of disbelief by providing alternatives that construct the narrative of the existence of those conceptual products oriented to generate doubt, and existential conflicts. In a way similar to how fashion designers work, they produce

collections of pieces they know they won't sell, but that helps to both market themselves and to explore new lines of thought.

Mogensen, the Danish scholar, with his »prototypes« idea talks about a very similar concept, about using elements, prototypes, that could trigger states of mind in the users (Mogensen, 1992). This is to reach the state of suspension of disbelief with a product in order to analyze how the test user copes with the tasks he is supposed to develop as a part of the experimental setting.

CONCLUSION: RESIGN DESSEARCH

In an exercise of creative freedom I have decided to create my own way of calling the area of design dedicated to the creation of methods. It is inspired by my practice as interaction designer, as member of the design collective Deseach and Development. In that case we deconstructed the generic idea of the department of innovation within companies, the so-called Research and Development department, by exchanging the first letter of each word. Then, the real or constructed meaning of the terms didn't matter, at least not in the moment of the creation. We have of course constructed a narrative around it over time. With that in mind, and with the experience of looking into the etymology of different words during the research needed for writing this text, I reverse engineered the expression »Resign Dessearch«, which becomes a pragmatic manifesto for a different kind of research for the design field, one that allows to still be playful, ideologist, and sometimes politically incorrect.

»Resign« is constituted by the prefix »re« meaning »to repeat«, »to iterate« and the root »sign« (signature, profile). Therefore, »resign« refers to the idea of »reaffirming the essence«, »noting those things of relevance.«

»Desearch« has the prefix »de« which is of a negative nature, negates the following term, which is »search«: »to look for«, »inquiry.« This is then a complex term, with a controversial meaning: the process of searching through introspective, sometimes destructive, but always unconventional and hands-on attitudes.

Bringing both concepts together, »Resign Dessearch« becomes the term that refers to reaffirming the essence of searching with unconventional methods.

Resign Dessearch (RD from now on) is an evolution of the latest school of thought, but like the other ones it has the property of not being destructive and coexisting with the other design methodologies, theories, and practices. Unlike other research disciplines, RD cannot afford to fail in a research process. In order to do research,

we have to make design, which will probably imply the study of people. The ideology of RD is not allowing it to fail when counting with people, either users or clients, as receivers of the design result. Therefore hypothesis testing is not a tool to be used, in order not to have an excuse to be right even if the goal is not achieved. RD has to learn both from the successes and from the failures, but then we cannot stop calling things what they are.

Again unlike traditional design research methodologies, RD is not having single results as outcome for projects. It will either propose to use already existing services, devices, and tools, or it will conclude with a »collection«. Collections are what fashion designers introduce at their catwalks. Those are characterized by some features:

- the designs are not necessarily meant to be produced, designers use them for promotion
- they allow to trigger a certain question or issue, don't need to be functional
- they are made in a quick way, and for models with standard sizes, allowing to work in an easy way simplifying the design process
- sometimes the collection is bought by an external actor, including the whole concept, the production process, research materials, etc

My proposal is then to think about interaction design in this same way. Nowadays, in the era of physical computing, we see the design of artifacts as a slow craft that requires such an effort that it becomes hard to create more than one type of artifact to compare. At the same time, there is a lot of research to be done in terms of cognitive psychology and the use of interfaces. But the day will come when instead of thinking about prototyping as putting electronic components together, we will take the interfaces for each cognitive nexus from a box and will attach them together on a device. Whenever that time comes, then we will be able of offering a whole collection of designs instead of only one at the time. Like fashion designers do, an interaction designer working in this way would prepare a show of pieces that could illustrate his visions on the topic inspiring the collection. The designer's customer is not the final user, but a certain intermediary. Therefore the designer's goal is to take a selection of basic interaction modes, layout tricks, cognitive psychology tricks, and put them in a device.

Slowly users are getting a better and better understanding of what it means to use a digital artifact, about the form-function disconnection, screen-centered interaction, etc. Therefore when working within the field of design research we should be able of introducing this as a variation in the fictional story of the design. With fictional story of the design I mean that, even if we reach a certain level of suspension of disbelief, users will notice at some point that there is a trick behind the

artifact. By adding the idea of using how the market works to the research process, we are not just contributing to select which devices are preferred by the users, but to give the story a bit better grounding.

In fashion, when working with collections, designers know that the model's bodies follow a limited set of sizes, which makes it easy for them to prepare the design and minimize the preparations to make on the pieces before the show. In the interaction design field that is what we call configuration, which should allow the designers to adjust the pieces very quickly to the user. But the problem here is to find the equivalent to the models in our field. The model has the characteristic of being someone in between the final user and the designer, someone with special characteristics who can test out how good the design is. Interaction design could work in a similar way, first designers could make a collection, and rough functional prototypes could be tested in different modalities with several »models« (beta testers). These testers, experts in showing, would then make an open show for the press and invited guests. From there, companies could choose whether to produce the pieces or some of them. This would allow playing with more complex aspects addressing the accumulation of functions. Nowadays most of the efforts of e.g. mobile telephony production are focused in the introduction of one or some small new features. The question is for how long this will last this way. I envision that the day will come when everyone in this business will work with collection as a basic way of designing, and we will evaluate more complex patterns of interaction.

One might think that I am leaving the ideological argument out of the discussion. On the contrary, as a designer I am trying to design the way design works, which means that I am following the idea of optimization of the production process that was leading the original design's ideology, the possibility of reaching everyone in terms of mass-production.

To me it is clear that we will need to collect the results, and analyze them, as Friedman writes, if we want to construct a line of thought coherent with contemporary established research disciplines, but that should not affect the »crafts« aspect of being a designer, because the craft, the artistic part of design is what makes it different from the other areas. If it is taken away, there will be no difference between design and sociology.

REFERENCES

- Chaves, N. (2001). *El oficio de diseñar*. Editorial Gustavo Gili.
Dunne, A., Raby, F. (2001). *Design Noir*. Birkhauser.
Flusser, V. (1999). *The shape of things*. Reaktion Books.

- Friedman, K. (2003). Theory construction in design research: Criteria, approaches, and methods. *Design Studies* 24.
- Gaver, W.H., Dunne, A., Farrington, P., Hooker, B. (2001). *The Presence project*. Art Books Intl Ltd.
- Hooker, B., Kitchen, S. (2003). Edgetown. <http://www.edgetown.net> (accessed September 9, 2004).
- Mogensen, P. (1992). Towards a provotyping approach in systems development. *Scandinavian Journal of Information Systems* 4:31–35.
- Zimmermann, Y. (1999). *Del diseño*. Editorial Gustavo Gili.



APPENDIX B, PUBLICATION 1

Appendix B, Publication 1

R. Casas, D.Cuartielles, Á. Marco, H. J. Gracia & J. L. Falcó. Hidden issues in deploying an indoor location system. *IEEE Pervasive Computing*, 6(2) (2007), 62–9.

© 2007 IEEE. Reprinted with permission from the authors.

Hidden Issues in Deploying an Indoor Location System

Installing indoor location system prototypes yields practical lessons about how to design and deploy future ubiquitous technologies.

The design of context-aware technologies has been on many research team agendas since Mark Weiser first described his ubiquitous computing vision. Determining the location of people and objects in indoor environments with a high degree of accuracy is a major technical obstacle to achieving this vision. We began development of the Bluetooth and Ultrasound Positioning System¹ in 1999 with the objectives

of high location accuracy, low cost, and minimal infrastructure requirements in healthcare environments. BLUPS uses Bluetooth and ultrasound to measure distances between mobile stations (MSs) and a base station (BS), whose location is known. We refined our location technologies during several design iterations and installations of BLUPS, from 2001 to 2005. The system's spatial accuracy, when measured empirically, exceeded our expectations.

Another pleasant surprise was the variety of system applications we discovered. At the beginning, we designed the system to help increase the autonomy of people with disabilities by detecting high-risk situations, enabling automatic guidance of wheelchairs, and providing orientation and guidance when entering public buildings. We installed such systems at Spanish organizations and universities at Zaragoza and Sevilla, and in Basque country. Beyond healthcare, we saw a potential use for BLUPS in the entertainment field, producing a framework for sound art installations that generate "soundscapes" at the

University of Potsdam and Malmö University, where visitors can explore sound-augmented spaces using a location-aware PDA. At Malmö University, BLUPS evolved into a research tool for studying how people use public spaces. These installations gave us practical and theoretical findings that are highly valuable in the design and deployment of an accurate indoor positioning system.

Over several BLUPS design iterations, prototype deployments, and system installations, we learned that it's almost impossible to anticipate the situations that can make an installation infeasible. The installation process itself became a priceless tool in evaluating how to deploy ubiquitous indoor technologies. Here, we collect these lessons learned regarding both technological and practical factors in the design and deployment of an indoor positioning system.

Lessons learned from controlled-environment deployments

After analyzing a variety of positioning technologies (see the "Picking a Positioning Technology" sidebar), we selected an RF-ultrasound combination to measure MS-to-BS distances in BLUPS. This technology coupling allows a subcentimeter precision using off-the-shelf electronic components.

As with GPS, if we can obtain accurate distances to positioned base stations, computing precise locations becomes a well-known mathematical problem. Unfortunately, this only happens under ideal conditions that are rare in real deployments. Because an exhaustive analysis of all potential deployment situations is impractical, we've categorized four error sources that arise when estimating MS-to-BS distances:

Roberto Casas
Technical University
of Catalonia, Spain

David Cuartielles
Malmö University, Sweden

Álvaro Marco, Héctor J. Gracia,
and Jorge L. Falcó
University of Zaragoza, Spain

Picking a Positioning Technology

Our work addresses indoor ubiquitous computing applications that require high spatial accuracy (below 5 cm), large coverage areas (up to building-sized areas), and low cost. Researchers have applied communication technologies to estimating location (for example, GSM¹ and Wi-Fi²). These technologies are appealing because they don't require proprietary hardware. So, they're inexpensive and widely used. However, even in ideal cases, these systems are accurate only to the range of several meters.

RFID achieves similar accuracy. Under ideal conditions, several existing RFID commercial systems achieve a precision of 1 meter in wide-area deployments (for example, see ActiveWave RFID solutions, www.activewaveinc.com).

In wide-area deployments, ultrawideband systems provide the best performance. Ubisense's system can achieve an accuracy in the range of tens of centimeters, largely because it can measure signal angles and differences in arrival times and because structures don't significantly affect its performance.³

Location systems that use ultrasonic radiation usually measure an ultrasonic wave's time of flight (TOF). Just as inertial sensors are frequently used in collaboration with other technologies, ultrasound systems often use RF or optical technologies to set a common temporal reference that helps determine the distance from emitters to receivers and thereby compute the position. Active Bat⁴ and Cricket⁵ employ this strategy, achieving an accuracy of several centimeters; however, they require a substantial infrastructure of sensors and transmitters.

The Constellation system⁶ uses infrared radiation to trigger the location process, ultrasound to measure the TOF from the emitters to the receiver, and inertial sensing to track these location processes. The system achieves millimetric accuracy, low latency, and a high refresh rate, but again requires a large network of transmitters.

Our Bluetooth and Ultrasound Positioning System's performance

can locate up to 30 mobile stations simultaneously at an accuracy of 3 cm and a refresh rate of 2 Hz. So, ours' performance is similar to Active Bat's and Cricket's but inferior to Constellation's. Nevertheless, its minimal infrastructure (five to eight fixed base stations per 100 m² versus 10 times as many for Constellation in the same space) can be a significant advantage in many situations.

Many options for providing a positioning service exist, but none is perfect. Application designers must prioritize their requirements to choose the most appropriate system. The most obvious considerations are accuracy, range, refresh frequency, and cost. The final decision must also consider the various issues that emerged from our experiments and installations, such as the amount of infrastructure, calibration process, total number of devices to be tagged simultaneously, and the system's robustness and immunity to occlusions.

REFERENCES

1. V. Otaso et al., "Accurate GSM Indoor Localization," *Proc. Ubicomp 2005*, Springer, 2005, pp. 141–158.
2. T. Kitasaka et al., "WPS: Location and Motion Sensing Technique of IEEE 802.11," *Proc. 3rd Int'l Conf. Information Technology and Applications (icta'05)*, vol. 2, IEEE CS Press, 2005, pp. 346–349.
3. J. Cadman, "Deploying Commercial Location-Aware Systems," *Proc. 5th Int'l Conf. Ubiquitous Computing (Ubicomp 03)*, LNCS 2864, Springer, 2003, pp. 4–6.
4. M. Addlesee et al., "Implementing a Sentient Computing System," *Computer*, vol. 34, no. 8, 2001, pp. 50–56.
5. N.B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," *Proc. Mobicom 2000*, ACM Press, 2000, pp. 32–43.
6. E. Foxlin et al., "Constellation: A Wide-Range Wireless Motion-Tracking System for Augmented Reality and Virtual Set Applications," *Proc. Siggraph 98*, ACM Press, 1998, pp. 371–378.

- **Estimation errors.** When ultrasonic radiation isn't blocked, you can easily estimate distance by using the chirp's time of flight (TOF). However, problems with synchronizing the BS and MS, calculating the speed of sound, and operating the ultrasound reception hardware can introduce estimation errors. We found a realistic estimation error of 1 percent in our deployments.
- **Lightweight obstruction errors.** We ran several blocking tests for cases in which people or objects around the

MS hindered the signal. We obtained a maximum error of 5 percent, even for crowded scenarios, but only when the blocking object wasn't very close to the MS. We saw a similar error when poor environmental conditions such as strong air currents or large temperature gradients seriously affected the speed of sound.

- **Non-line-of-sight errors.** When the signal blockage is severe—for example, a person or object is in the BS's line of sight at a short distance from the MS—the error is termed non-line-of-sight. The

NLOS error's magnitude depends on the object's size, position, and composition and is likely to be somewhere between 10 and 100 percent of the true distance.

- **Full blockage errors.** The worst situation occurs when the BS's signal is completely blocked. A completely wrong TOF measurement can also result from hardware or RF synchronization problems.

The application's influence on system architecture

Distances can be measured from the BS

LOCATION SYSTEM DESIGN

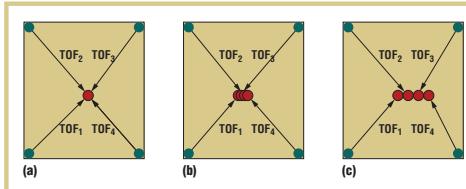


Figure 1. Time-of-flight measurement of a mobile station as it moves at different speeds: (a) static location; (b) slow moving; and (c) fast moving. The transmissive base station architecture can track a large number of MSs but loses accuracy when an MS speeds up.

to MS or vice versa. Accordingly, two architectural possibilities exist: if the MS emits ultrasound signals, we call it the receptive base station (RBS) strategy (as used by Active Bat²), whereas if the BS is transmitting, the architecture is the transmissive base station (TBS) strategy (as used by Cricket³ and Constellation⁴). Neither approach has a general advantage over the other; the suitability of each system depends on the physical environment, type of user, and field of application.

The first BLIPS deployment covered a single room of 100 m² and lasted two years. Because the application involved residences for people with mental and physical disabilities, the system needed to intensively track people to detect activities such as short, repetitive routines that might denote, for example, anxiety or desire to escape. The basic sys-

tem aims were robustness and accuracy. We knew that blockages at some distance from the ultrasound receivers have much smaller propagation effects than those that occur closer to the emitters. Thus, considering that objects and people blocking the signals would most likely be closer to the MSs, we adopted the TBS architecture, placing the emitters close to the ceiling. This architecture also enabled the simultaneous positioning of a large number of users, because the BS sends the chirps.

In a three-year research collaboration with the University of Seville, we implemented the system in a larger space (150 m²) to guide an automatic wheelchair. The system had to provide information on the moving wheelchair's position at the highest possible rate and accuracy. We quickly identified the TBS

scheme's limitations in estimating a location for the moving MSs. The entire system operates on a single ultrasound band, so multiple BSs can't emit chirps simultaneously because the MS can't distinguish the received signals' origin. We must sequence BS emissions so the moving MS can receive each ultrasonic chirp (corresponding to different BSs) at different positions. The distances obtained this way don't correspond to a unique location, so the employed algorithms generate larger errors as MS speeds increase, as figures 1a through 1c show.

In the end, we decided to use the RBS architecture because it provides better accuracy when locating mobile devices. Moreover, instead of a single ultrasonic propagation time for each BS, we only need one per MS. The RBS architecture also increases the location refresh frequency.

We're also using this architecture in a different project that tracks a surgical instrument for transurethral resection

Figure 2. Surgical instrument for transurethral resection, showing (a) the instrument in use and (b) the location of the points we use to calculate spatial orientation. The receptive base station architecture maximizes accuracy and refresh rate for a small number of trackable objects.

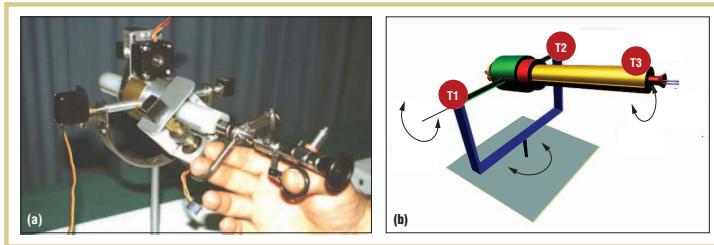


Figure 3. Performance of the least-squares method versus least-median-of-squares (LMedS) method. All the measurements have a random uniform error with a maximum amplitude of 1 percent. In addition, n measurements have a non-line-of-sight (NLOS) error with a maximum amplitude of 50 percent.

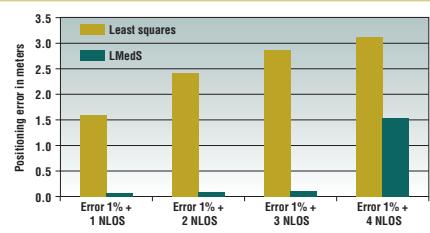
(figure 2a). In this project, we must locate three points on the instrument—T1, T2, and T3—to orient the instrument in a space of 1m^3 (figure 2b). Again, it is critical to achieve the highest accuracy and fastest refresh rate possible.

In short, we discovered that the RBS strategy is the most appropriate architecture when you must intensively track a few mobile devices.

The positioning algorithm's importance

Once we've obtained the distance from an MS to each BS, we can calculate the MS location. Researchers have developed many strategies to solve this problem, including neural networks, iterative methods, algebraic techniques, and statistical procedures. All these approaches are susceptible to the four categories of blocking errors we described earlier. We've studied this problem exhaustively,² but owing to its complexity and specialization, we present only the conclusions that apply most to real scenarios.

If the starting data includes no significant errors, almost any algorithm will achieve a high degree of accuracy. However, if most distance estimates are corrupted, any algorithm will fail because no valid relationship exists between the starting data and the output. In these situations, only a priori knowledge of the system's behavior will help. Outdoor scenarios like GSM location can use statistical data about signal errors to improve estimate accuracy. The equivalent for indoor environments—performing statistical analysis of ultrasound propagation—doesn't make sense, however, because the conditions change quickly and unpredictably. Usually, the initial data comprises a mix of correct and



erroneous data. In such cases, we might have enough data to calculate the position accurately (three distances in three dimensions), but erroneous data prevents us from reaching the solution directly if we consider all the data at once. This is common for indoor positioning systems: the BSs are distributed around a room, and some of them—we don't know which—have a direct view of the MS, while others are hidden by the environment.

We found the least-median-of-squares (LMedS) algorithm to be best in situations that lack prior information for solving the multilateration problem.³ The LMedS algorithm, far from simultaneously using all available data, compiles sets of input measurements that it uses to calculate a space of solutions and estimate the residuals of all the measurements. The so-called robust solution is the one with residuals that have the minimum median. The "Least-Median-of-Squares Algorithm" sidebar on page 66 details this algorithm.

Figure 3 compares the LMedS algorithm to least-squares, the method most commonly used to solve nonlinear systems that converge to a global minimum. We consider the case of a $10 \times 10 \times 5$ meter installation with eight BSs where between one and four distance measurements are affected by heavy NLOS error (50 percent). The LMedS algorithm can manage the NLOS errors effectively, whereas the least-squares algorithm fails, even with just a single erroneous distance. LMedS can handle up to three measurements with NLOS

errors out of a total of eight measurements—a proportion of errors quite unlikely in a real situation.

The most important conclusion of our analysis is that the system must collect enough correct data to accurately calculate the MS's location, regardless of the environment. Therefore, the best way of ensuring accurate positioning is to collect redundant data and use a good algorithm that can filter out erroneous measurements. We can collect more data by setting more BSs than should be necessary, but how many do we need? The clearest evidence of the algorithm's importance lies in the relationship between accuracy and the BS density our system requires versus the density other systems require. When using LMedS, we can simultaneously locate up to 30 MSs at an accuracy of 3 cm and a refresh rate of 2 Hz with less than one BS for every 10 m^2 (eight BSs in a 100 m^2 installation) to obtain an accuracy of several centimeters, even under adverse conditions. All other systems require a much higher density of BSs to achieve the same accuracy; for example, Active Bat requires 750 sensors in a 10,000-square-foot office building.² The difference is significant because reducing the infrastructure's complexity ensures a more cost-effective system and easier deployment.

Lessons learned from real location system deployments

We applied the test scenario lessons to several real-world BLUPS installations.

Least-Median-of-Squares Algorithm

The least-median-of-squares (LMedS) algorithm is a robust method widely used in artificial vision problems.^{1,2} If the objective is to fit data into a model instead of using all the available data, LMedS makes several subsets with parts of the data to obtain possible solutions for that model, and chooses the one that minimizes the median of the residuals.

If some, but not all, of the data is corrupted, some subset without erroneous data could exist that would give a more accurate estimation for the model. We can apply this idea to perform localization with the following algorithm.

- Because we require a minimum of three base stations (BSs) to compute a mobile station's (MS) location, and a total of n BSs exist, we make m subsets of three distances:

$$m = \frac{n!}{3!(n-3)!} \quad (1)$$

- For each subset S_i of distances (d_u, d_v, d_w) , we compute a location $P(x, y, z)$ by solving the following systems of equations using any traditional techniques, such as least squares:

$$\begin{aligned} d_u &= \sqrt{(x_i - x_{BSU})^2 + (y_i - y_{BSU})^2 + (z_i - z_{BSU})^2} \\ d_v &= \sqrt{(x_i - x_{BSV})^2 + (y_i - y_{BSV})^2 + (z_i - z_{BSV})^2} \\ d_w &= \sqrt{(x_i - x_{BSW})^2 + (y_i - y_{BSW})^2 + (z_i - z_{BSW})^2} \end{aligned} \quad (2)$$

Again, the installation process revealed its own lessons in how to deploy future ubiquitous indoor technologies.

System configuration

A key issue when installing a location system in a real environment is how to ensure good performance over the entire deployment area. For all positioning systems, this means providing proper signal coverage—that is, determining the appropriate number of BSs and their optimal locations.

Probir Kumar Ray and Ajay Mahajan used genetic algorithms to obtain the optimal BS distribution.⁶ However, following their recommendations leads to impractical placements. Specifically, you must place some BSs at floor level, significantly blocking their chirps and ren-

dering them useless. According to our earlier experiences, the best positions are those sites that maximize the number of NLOS-free BSs. Thus, to maximize algorithm performance, the best sites should provide maximum ultrasonic coverage of the area in which the MSs are located.

Before calculating the appropriate number and optimal positions of BSs, we must characterize the devices ultrasonically. Ultrasound performance is strongly influenced by electronic conditioning hardware, and choosing the best transducers possible is the most relevant design decision. Saeed Shiry Ghidary and colleagues analyzed in detail how to mount transmitters and receivers to ensure the desired coverage in terms of transducer sensitivity, sound-pressure level, and beam angle.⁷ However, they

performed their analyses only under ideal conditions. To optimize infrastructure for a scenario, we performed an orientation and blocking analysis to determine potential error sources. Once we know the behavior when blocked and the 3D range that our BSs and MSs can achieve, we can ensure that the BSs (in the case of RBS) or the MSs (for the TBS strategy) receive sufficient ultrasonic chirps to compute the location of MSs. Our system requires at least three BSs, but in adverse environments, such as heavily populated areas, five to eight BSs provide a more robust positioning.

BLUPS uses RF to provide communication and synchronize the system. Ensuring radio coverage involves a process similar to determining the access point positions when deploying a wireless

where $i = 1, \dots, m$ and (x_{BS}, y_{BS}, z_{BS}) are the coordinates of BS $_j$ and $j = u, v, w$.

- For each location P_j , we obtain the residues R_j as

$$R_j = \left((d_1 - \hat{d}_{j1})^2, (d_2 - \hat{d}_{j2})^2, \dots, (d_n - \hat{d}_{jn})^2 \right) \quad (3)$$

where

$$\hat{d}_{jk} = \sqrt{(x_j - x_{BSk})^2 + (y_j - y_{BSk})^2 + (z_j - z_{BSk})^2}, \quad k = 1, \dots, n \quad (4)$$

and compute the median M_j of the residues R_j .

- The final solution $P(x, y, z)$ is that with the minimum median M_r .

REFERENCES

- Z. Zhang, *Parameter Estimation Techniques: A Tutorial with Application to Conic Fitting*, tech. report 2676, Inria, Oct. 1995.
- D. Mintz, P. Meer, and A. Rosenfeld, "Analysis of the Least Median Of Squares Estimator for Computer Vision Applications," Proc. IEEE Conf. Computer Vision and Pattern Recognition, IEEE CS Press, 1992, pp. 621-662.

Figure 4. Coverage area and position of BSs in a multicell installation. (a) The installation covers a wide entrance hall and cafeteria of 1,000 m². (b) The base stations labeled 1, 2, 3, and 4 transmit beams at angles of 90, 180, 270, and 360 degrees.

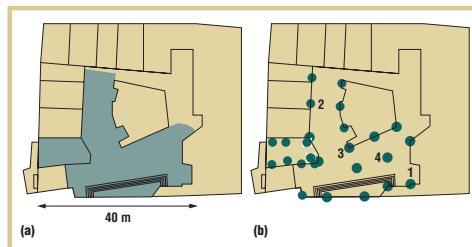
communication system such as Bluetooth or Wi-Fi. Each building has its own characteristics, and the people who install the system must empirically determine the path loss and wave propagation characteristics.

Figure 4a shows the coverage area for the first multicell installation we undertook at Malmö University. For six months, we covered a 1,000 m² area, 5 m high, with 26 BSs. We knew the scenario would be relatively crowded with people and would involve moving furniture on occasion. Consequently, we decided to ensure MS coverage from 5 to 6 BSs at every point. After characterizing the range and beam patterns of the BSs and the MSs, we decided on the BS locations in figure 4b. The installation used the TBS strategy, and we treated the BSs as if they were GSM stations, where the repeating equipment (the BS electronics) can connect to different antenna types (the ultrasound transducer). Thus, we used four kinds of BSs, each sending our beams at different angles. In figure 4b, the BS labeled 1 covers 90 degrees, 2 covers 180 degrees, 3 covers 270 degrees, and 4 covers 360 degrees. We positioned all BSs close to the ceiling, ensuring robust coverage beneath them.

System calibration

Location estimation involves the calculation of ranging distance and position. Calibration ensures proper functioning of these two tasks.

An accurate estimate of the speed of sound is important when measuring distances. During normal operation, we just use air temperature to estimate the speed of sound in each room. Other factors either have much less influence (in the case of humidity) or have a nondeterministic



influence (in the case of air turbulence).

To calculate the optimal MS positions, we need the BSs' coordinates; the accurate estimation of these coordinates is a major task in real deployments. Directly measuring the 3D distances to dozens of BSs (placed several meters above the floor) from a reference point at centimetric precision is complicated and laborious using 1D positioning systems such as measuring tape (figure 5). A method widely used in wireless sensor networks lets BSs emit and receive ultrasound signals and measure the distances between them.⁸ Thus, once three BSs are manually positioned, the rest can calculate their own position. Unfortunately, this approach only works in a single cell. Because ultrasound requires direct line of sight, doors and walls force us to repeat the process in every cell. Moreover, you have to endow the BSs with both ultrasound emission and reception hardware. As an alternative, we found a practical method from Ajay Mahajan and Fernando Figueroa,⁹ who proposed inverting the MS and BS roles. In this method, you must move the MSs to known locations to measure the TOF from the BSs. By gathering information from at least five MSs locations, you can calculate the BSs' positions using the proposed algorithms. Mahajan and Figueroa tested this method in small installations (up to 8 m²) with good results. Nevertheless, we found an important drawback for large installations. Although the error in deployment areas that are 1.3 m across was just 1 percent, the errors for an area that is 10 m

across increase beyond 10 cm, which is unacceptable for high-accuracy systems.

We improved this procedure in several regards. First, we customized the algorithms. In addition to the BS coordinates, Mahajan and Figueroa's algorithms introduce the speed of sound and fixed time delays as unknown variables. Using the same data, we first calculate these two parameters. Then, applying the LMedS algorithm, we calculate the BS positions as if they were MSs. In this way, we need fewer calibration points—three to five—and achieve a higher estimation accuracy. Examining the results, we realized that the horizontal coordinates' estimates were notably more accurate than the vertical coordinates' estimates. Analyzing the problem's geometry, we concluded that this is because the BSs are installed to ensure ultrasonic coverage rather than to optimize the algorithm behavior.^{5,6} Distributing all BSs in the ceiling so that they have a similar Z coordinate is not good in terms of estimating the vertical coordinate. To overcome this limitation, the best solution is to measure Z manually. This is not a large inconvenience because X and Y are by far the most problematic measurements. The procedure required to accomplish a 0.1 percent accuracy in positioning the BSs in each cell is, in summary:

- from four positions spread over the cell, calculate the ultrasonic TOF to the BSs (three are sufficient, but one more confirms the absence of errors);



LOCATION SYSTEM DESIGN



Figure 5. Installing base stations. Obtaining accurate positioning data is difficult when you use actual measuring tape rather than algorithmic calculations.

from days to hours. Simplifying the procedure will also enable nonspecialized personnel to help set up the system, which brings the process closer to a real-life situation.

Batteries versus wires

A location system's setup type and application determine the source used to power it. Our first prototype aimed for a completely wireless, battery-driven, autonomous system. We considered neither power consumption nor battery life in our initial designs. We believed we could solve such issues in the long run using an appropriate power supply. The first version of BLUPS used lead batteries for the BSs and two AAA rechargeable batteries for the MSs.

In the prototype's second iteration, we realized that we didn't need to power the BSs with batteries in fixed setups. We then decided to wire all BSs, making use of a single power supply. In addition to eliminating the chore of periodically replacing BS batteries, this improved the system's calibration, quality, and lifetime. Because BSs are attached to the wall, it's difficult to replace the batteries while keeping the ultrasound transceivers in their original position. For this reason, before we'd decided to wire all BSs together, we'd already equipped them with several external cables that took electricity from the batteries into the electronics.

- measure the BSs' distance from the floor;
- calculate the circuitry's speed of sound and fixed time delays; and
- compute and verify the BS's coordinates.

Of course, to obtain good results, you must perform the calibrations under the best possible conditions (that is, no ultrasonic shadows or strong air currents).

Algorithms that are more optimized and that involve simple calibrations ensure an easy, fast system deployment. For a single cell with six BSs placed 4 m above the floor, we reduced the calibration time from 1 hour to 5 minutes. This will minimize costs because it will reduce the time to adjust a large installation

to a mixture of wired and wireless artifacts. Because all the BSs already had wires, we redesigned the wires to communicate over RS-485. The power line lengths resulted in voltage losses, so we switched to a power supply mesh with separate power sources grounded together. Using bus transceivers instead of Bluetooth chipsets obviously saved money; by eliminating the Bluetooth transceiver, we lowered the cost of each BS from around €25 to €15. In addition, using a wired communication backbone reduces the complexity synchronizing and managing the Bluetooth network requires. Tasks such as roaming (MSs moving from piconet to piconet) and periodic analysis of new device arrivals into the system become greatly simplified. Moreover, installing this system doesn't differ significantly from wiring a series of connectors for telephony, a job that an electrician or network installer can perform.

However, we never abandoned the idea of constructing a completely wireless system that can be deployed quickly.

We officially presented one at the Participatory Design Conference (Toronto, 2004) as an example of the application of location technologies to art.¹⁰

In essence, location technologies are a collection of artifacts that are attached to humans, objects, or animals (the "carrier") to enable someone to estimate the carrier's location via communication with a certain type of infrastructure. BLUPS combines Bluetooth and ultrasound to provide a high-accuracy indoor positioning system at a reasonable cost. In our work so far, we've identified several areas in which we can evolve it.

One area for improvement is the communications technology. To reduce power consumption and ease the roaming requirements of the mobile devices, we're currently working to replace Bluetooth

the AUTHORS



Roberto Casas is an assistant professor in the Technical University of Catalonia's Electronic Engineering Department and a member of the Technologies for Disability (TecnoDiscap) group at the University of Zaragoza. His research interests include sensor networks, digital electronics, and assistive technology. He received his PhD in electronic engineering from the University of Zaragoza. Contact him at Dpto Electronica, Avd. Canal Olimpic SN, 08860, Castelldefels, Spain; rcasas@eel.upc.edu.



David Cuartielles is the head of the Prototyping Laboratory at Malmö University's School of Arts and Communication, while he conducts his doctoral thesis analyzing the boundaries between technology, arts, and society. He received his MSc in telecommunications engineering from the University of Zaragoza, where he was also a member of the Technologies for Disability (TecnoDiscap) group. Contact him at K3, School of Arts and Communication, Malmö Univ., Enh. 4, SE 20506, Sweden; david.cuartielles@k3.mah.se.



Álvaro Marco is an assistant professor in the University of Zaragoza's Computer Science and Systems Engineering Department and a member of the Technologies for Disability (TecnoDiscap) group at the University of Zaragoza. His research interests include sensor networks, digital electronics, and assistive technology. He received his MS in electrical engineering from the University of Zaragoza. Contact him at Dpto Electronica y comunicaciones, C. María de Luna 3, 50018, Zaragoza, Spain; amarco@unizar.es.



Héctor J. Gracia is researcher in the University of Zaragoza's Electronic and Communications Department and a member of the Technologies for Disability (TecnoDiscap) group at the University of Zaragoza. His research interests include analog and digital electronics and assistive technology. He received his MS in electrical engineering from the University of Zaragoza. Contact him at Dpto Electronica y comunicaciones, C. María de Luna 3, 50018, Zaragoza, Spain; hgracia@unizar.es.



Jorge L. Falcó is an associate professor in the University of Zaragoza's Electronics and Communications Department and a member of the Technologies for Disability (TecnoDiscap) group at the University of Zaragoza. His research interests include electronic instrumentation and assistive technology. He received his PhD in electronic engineering from the University of Zaragoza. Contact him at Dpto Electronica y comunicaciones, C. María de Luna 3, 50018, Zaragoza, Spain; jfalc@unizar.es.

ACKNOWLEDGMENTS

The European Union partially supported this work under the project IST-5-053147; the Spanish Ministry of Science and Technology also partially supported it under CICYT project numbers TIC2003-07766 and TIN2006-15617-C03-02.

REFERENCES

1. R. Casas, "BLUPS: Bluetooth and Ultrasound Positioning System," doctoral dissertation, Univ. of Zaragoza, 2004.
2. M. Addlesee et al., "Implementing a Sensoriented Computing System," *Computer*, vol. 34, no. 8, 2001, pp. 50–56.
3. N.B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," *Proc. Mobicom 2000*, ACM Press, 2000, pp. 32–43.
4. E. Foxlin et al., "Constellation: A Wide-Range Wireless Motion-Tracking System for Augmented Reality and Virtual Set Applications," *Proc. Siggraph 98*, ACM Press, 1998, pp. 371–378.
5. R. Casas et al., "Robust Estimator for Non-Line-of-Sight Error Mitigation in Indoor Localization," *Eurasip J. Applied Signal Processing*, vol. 2006, article ID 43429, 2006, pp. 1–8.
6. P. Kumar Ray and A. Mahajan, "Optimal Configuration of Receivers in an Ultrasonic 3D Position Estimation System by Using Genetic Algorithms," *Proc. American Control Conf.*, IEEE Press, 2000, pp. 2902–2906.
7. S.S. Ghidary et al., "A New Home Robot Positioning System (HRPS) using IR Switched Multi Ultrasonic Sensors," *Proc. IEEE Conf. Systems, Man, and Cybernetics*, IEEE Press, 1999, pp. 737–741.
8. H. Balakrishnan et al., "Lessons from Developing and Deploying the Cricket Indoor Location System," tech. report, MIT, 2003.
9. A. Mahajan and F. Figueiroa, "An Automatic Self Installation & Calibration Method for a 3D Position Sensing System Using Ultrasonics," *Robotics And Autonomous Systems*, vol. 28, no. 4, 1999, pp. 281–294.
10. D. Cuartielles, "Embodied Sound," *Proc. 8th Participatory Design Conf.* (PDC 04), vol. 2, Computing Professionals for Social Responsibility, 2004, pp. 16–18.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.



APPENDIX C, PUBLICATION 1

Appendix C, Publication 1

D. Cuartielles & D. Taylor. Delivery number D2.1: *Datasheets for SandS Motherboard and Modules* (Malmö: Social&Smart, 2013b) [SandS].



SandS: Social & Smart

Social housekeeping through intercommunicating appliances
and shared recipes merged in a pervasive web-services
infrastructure

Delivery number D2.1

Datasheets for SandS Motherboard and Modules

Project	
Project acronym:	SOCIAL&SMART
Project Full title:	Social housekeeping through intercommunicating appliances and shared recipes merged in a pervasive web-services infrastructure
Grant agreement no.:	317947
Document	
Deliverable number	D2.1
Deliverable title:	Datasheets to SandS Motherboard and Modules
Workpackage:	Wp2
Due date of deliverable:	September 2013
Lead beneficiary:	ARD
Editors:	David Cuartielles, Davey Taylor
Contributing beneficiaries:	ARD
Reviewer:	Gian Luca Galliani
Status:	Final draft
Version and date	V0001, 26/09/2013
Changes	Added Firmwares inside zipped file

Project co-funded under the SEVENTH FRAMEWORK PROGRAMME THEME 3 ICT -
INFORMATION AND COMMUNICATIONS TECHNOLOGIES

Dissemination Level		
PU	Public	PU
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the Consortium (including the Commission Services)	
CO	CO Confidential, only for members of the Consortium (including the Commission Services)	

Project co-ordinator: Prof. Bruno Apolloni

Università degli studi di Milano

Tel: +39 02 50316284

E-mail: apolloni@di.unimi.it

Project website address: www.sands-project.eu

Deliverable 2.1 September 2013



1. Publishable summary (ARD)

The scope of deliverable 2.1 is to create a series of datasheets for the different hardware designs created for the SandS project members to prototype connected appliances. These designs are part of the work made by Arduino during Workpackage 2 and set the ground for the different partners to understand how the different electronic blocks can be used as part of SandS work.

This deliverable will be complemented by D2.2, which will focus more on the software that can be created to control different types of appliances, as well as in a series of practical documents to help the partners getting started in using the modules described in D2.1.

This deliverable is a compendium of:

1. different datasheets for each one of the modules created
2. a datasheet for the SandS motherboard
3. a document describing the communication protocol between the modules
1. a folder with the firmware created for all the modules



CONTENTS

Deliverable 2.1 September 2013	1
Declaration by the project coordinator.....	2
1. Publishable summary (ARD).....	3
CONTENTS.....	4
2. Modules created.....	5
3. SandS Motherboard.....	5
4. Communication protocol.....	5
5. List of documents.....	6



2. Modules created

The different cases that can be found when trying to build a connected appliance require a reduced set of modules. Mainly we need to:

- read a sensor
- control a motor
- regulate a heating/cooling element
- measure the device's consumption
- display information to the user
- gather simple information from the users

All of those functions can be gathered in a set of four boards:

- a board with a sensor, to collect data from different types of sensors (resistive, capacitive, etc) with an on-board digital amplifier
- a board with a relay to turn on/off different high voltage parts
- a board with a mosfet transistor to get different parts to pulse using PWM signals
- a board with an RMS-to-DC converter to be able of estimating the appliances' use of current, as well as power consumption
- a UI (User Interface) board with a display and a couple of controllers – rotary encoder and touch switches – to allow user interaction with the device

The main characteristics to each one of those devices is explained as part of the different datasheets accompanying this document.

3. SandS Motherboard

The SandS Motherboard is a basic communication board able of talking to the internet via wireless communication. In essence it is a microcontroller board including a WiFi chip. The WiFi module, called DogStamp, is a self contained mini-board offering a serial port for the microcontroller to send and receive data. The board comes with a RS-232/485 converter, a SandS-I2C port (an normal I2C port with extra identification and alarm pins), and a serial port connector for the external UI module.

4. Communication protocol

As described in the TWI Protocol document coming with the deliverable, we have created a special revision of the I2C protocol that allows the different modules to:

- get automatically assigned identification numbers
- send alarms to the other modules, as well as to the motherboard to cease operations due to a system malfunction

In order to achieve this, we added two extra pins to the standard 4 pin connector of the I2C protocol (PWR, GND, SDA, SCL). It should also be noted that the modules can operate under the standard I2C protocol without having to run any kind of software reconfiguration.

5. Specific SandS contributions

The work performed by ARD for deliverable D2.1 has been focused in creating a module that should accommodate the different scenarios described in the previous theoretical project documents.

The Dogstamp module developed in collaboration with the external entity DogHunter, is based on a design that is becoming a standard in the world of embedded electronics. It consists in a module with the ability of performing multiple networks operations over WiFi or Ethernet and it is built on a specific processor from Atheros that internalizes all those operations. In that sense, the real novelty on the design of the SandS motherboard is the creation of a



microcontroller board that allows connecting potentially any device to the network, but also having a relatively big amount of memory space to implement drivers to different devices in it.

In a way, the technology implemented inside the DogStamp is similar to the one implemented by ARD into it's commercial product the Arduino Yun. However, the Arduino Yun and the SandS motherboard cover different areas and are oriented towards different audiences. One could say that the SandS motherboard is more of a "professional" design as it is made to accommodate the modules designed for the project, while the Arduino Yun, has less computing power on the microcontroller side and it is oriented towards the average Arduino audience, centered in the easy of use for people to easily connect anything to the internet.

Also the software written for the SandS motherboard is expected to be entirely different from the one made for Arduino Yun. The SandS motherboard is meant to become an MQTT proxy to send and get data to and from the SandS network infrastructure. The Arduino Yun comes with a piece of software called "bridge" that essentially makes very easy to make transactions with the net via http queries to a web application. This is entirely different from the SandS approach, where we will work using a pub/sub strategy for transmitting data between the devices and the social network of appliances.



6. List of documents

The following table describes the documents being part of D2.1.

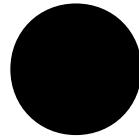
TABLE 6.1 LIST OF DOCUMENTS PROVIDED WITH D2.1

WP	Item name	Description	Filename
2	Relay	Module	201309_Arduino_Sands_Deliverable_2-1_Relay.pdf
2	Mosfet	Module	201309_Arduino_Sands_Deliverable_2-1_Mosfet.pdf
2	Sensor	Module	201309_Arduino_Sands_Deliverable_2-1_Sensor.pdf
2	Consumption	Module	201309_Arduino_Sands_Deliverable_2-1_Consumption.pdf
2	UI	Module	201309_Arduino_Sands_Deliverable_2-1_UI.pdf
2	Motherboard	Board	201309_Arduino_Sands_Deliverable_2-1_Motherboard.pdf
2	TWI-protocol	Software	201309_Arduino_Sands_Deliverable_2-1_TWI-protocol.pdf
2	Firmwares	Software	201309_SandS_Modules_Firmwares.zip

Document Information

Project reference

317947, FP7-ICT-2011-8



Context

Deliverable 2.1, workpackage 2

Document title

201309_Arduino_Sands_Deliverable_2-1_Motherboard

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

4. Oct. 2013

Table of contents

SandS Project.....	1
Document Information.....	2
Table of contents.....	3
Summary.....	4
SandS Motherboard.....	5
Embedded linux core.....	5
AVR core.....	5
Power the board.....	5
Hardware.....	6
Board dimensions.....	7
Mounting considerations.....	7
Schematic.....	7
Board file.....	7
Pin layout.....	7
Firmware.....	8
Open WRT.....	8
Licenses.....	9
Hardware.....	9
Firmware.....	9
Linux distro.....	9
Appendix 1 - Schematics.....	10

Summary

This document describes the main characteristics of the SandS motherboard, an open source hardware - Arduino compatible board with WiFi capabilities, exposing an RS-232/485 port, a UI-controller port, and a SandS-enhanced I2C port.

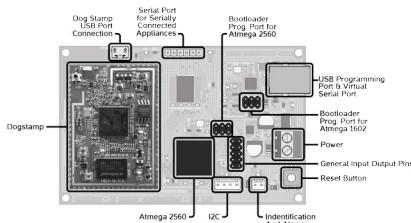


Image 1: block description, motherboard

The board runs from an external power supply, typically between 9V and 12V, can be reprogrammed from the Arduino IDE using a special library and offers the possibility of accessing the hardware remotely via a standard wireless connection.

Even if the board has been specifically designed for the SandS project, the pins exposed directly from the main processor (the ATmega2560) can be addressed as Arduino digital pins.

There are different on-board LEDs that can be used for debugging purposes since they can be accessed via software.

The communication between the main processor and the Linux core is done via serial communication. As the ATmega2560 has 4 internal serial peripherals, it uses one of them for the USB-Serial communication (mostly for reprogramming and debugging), one to talk to the Linux core, one to communicate with the external SandS UI controller, and the last one for the RS-232/485 driver.

SandS Motherboard

The SandS Motherboard is a microcontroller board with WiFi capabilities. It is made of two blocks:

- an embedded linux core based in a processor from Atheros, integrated in a module by Doghunter¹: dedicated to handle the wireless communication as well as different aspects of the SandS protocol
- a real time communication driver based on the ATmega2560 by Atmel²: this module is meant to handle all the communication at low level with serial, SPI, I2C (TWI) and USB ports

It comes with two connections for SPI, one I2C (two-wire interface), one RS-232 port, and one virtual serial port over USB. It also has two pins to be used as identification (**IDENT**) and alarm (**ALARM**) pins for the SandS implementation of the I2C protocol supporting both self-identification of the modules as well as reporting alarms to stop any operations on an appliance (see Figure 1 for a detailed diagram of the pins on the board).

Embedded linux core

The linux core handles the communication over WiFi towards the SandS infrastructure. It communicates through a serial port with the AVR microcontroller (see schematics, pins labelled DOG_RX and DOG_TX).

It is also possible to communicate with the processor both through the board's micro-USB connector and through wireless communication.

The basic characteristics of the embedded linux core used in this project are described in Table 1:

Processor	MIPS 24K operating at up to 400 MHz
Memory	DDR2 64MB Ram and 16 MB SPI Flash
AP or router	Complete IEEE 802.11bgn 1x1

Table 1: characteristics embedded linux core

The linux core comes as a self-contained module named Dogstamp. It runs a version of OpenWRT, a minimal linux distribution used on embedded devices.

¹ Doghunter is a Boston-based automation development firm dedicated to the creation of linux embedded devices

² Atmel: see <http://atmel.com>

AVR core

The AVR core chosen for the SandS Motherboard is the ATmega2560, which main characteristics are described in Table 2:

Processor	ATmega2560
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEROM	4 KB
Operating Voltage	5 V
Hardware Serial ports	4
Hardware SPI ports	1
Hardware I2C ports	1

Table 2: main characteristics of the AVR core

This processor can be programmed as an Arduino Mega 2560 board directly from the standard Arduino IDE, there is more information how to program the board on the *Getting Started* appendix to this datasheet.

The ATmega2560 does not have an internal USB peripheral, therefore, the SandS motherboard uses yet another processor from Atmel, the ATmega16U2, as a way to achieve USB communication.

Power the board

Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Current (typ)	TBD

Table 3: powering the board

During programming operations, the USB port can provide enough current to power up the AVR core. However, for the linux core to work, external supply is needed at all times.

Therefore, we recommend using a power supply within the power limits of the board.

Hardware

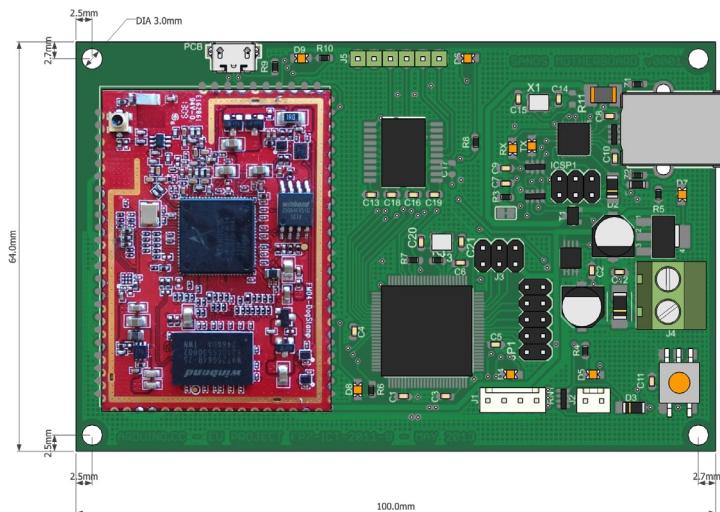


Image 2: Top view, SandS motherboard v0001



Image 3: Isometric view of the SandS motherboard v0001

Board dimensions

The board has been designed to easily accommodate the parts in a 2-layer design in the size of 64x100mm. It comes with 3mm holes at each corner, making it easy to mount it using standard plastic spacers and M3 screws.

Check Image 2 for further details about measurements and location of the holes on the board.

Image 3 gives you an idea of the proportions of the different components on the board.

Mounting considerations

It is recommended to mount the board in a plastic box. The WiFi chip requires no additional shielding. Furthermore, a metallic box could provoke bad wireless reception on the board.

The different modules will connect to the board using standard board-to-cable connectors and therefore it is recommended leaving space for those cables to exit the box. Alternatively it is possible to mount different types of connectors on the plastic housing and simply extend the board with those.

For safety purposes, this board's power input is through a screw connector. If you were in the need to use a different type of connector, you should fix it to the board's box and plug it in to the screw connector.

If you were about to add an extra power button to the board, you should do it by serially connecting it between the power connector on your box and the screw one.

The extra microUSB connector is not enabled in the motherboard configuration, therefore you don't need to fix a mounting hole for it on the box.

The standard-size USB-B connector is only used for reprogramming and debugging purposes, for a final installation, you could as well leave that one disconnected.

Schematic

The board's schematic shows all the components on it. You can check all the schematics at Appendix 1, at the end of this document.

The schematics have been divided into four pages:

- Power and connectors: shows the different pins and connectors on the board coming from the AVR core, as well as the serial-transceiver
- USB download: includes the ATmega16U2, its SPI connector for reprogramming the processor, the TX/RX LEDs, and the USB-B connector to plug in the cable
- Microcontroller: is the ATmega2560, the 16Mhz

crystal, the SPI connector for reprogramming the processor's bootloader, the reset button, and the connector for the UI board

- Dogstamp module: shows the Dogstamp module and its connections

Board file

The board file is a 2-layered design with the components separated strategically to avoid potential interferences. Half of the board is reserved for the wireless communication module, while the other half is dedicated to the AVR core, the RS-232/485 driver chip, the USB-serial conversion block, the power supply and the different connectors for the I2C, SPI and identification and alarm.

Images 2 and 3 give a pretty good idea of this layout. Both schematic and board file for this design come bundled in the same zipped file as this document and can be modified using the Eagle Cad software.

Pin layout

Image 4 shows the pin layout for the board as well as the description of each one of the pins separately.

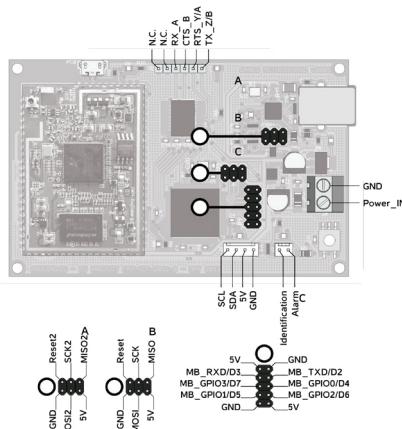


Image 4: pinout for the Sands Motherboard

Firmware

The SandS motherboard comes with two microcontrollers, and therefore a distinction must be made between each one of them when it comes to the firmware:

- The ATmega16U2 is responsible for the USB-Serial communication, it implements the standard USB-Serial conversion firmware by Arduino. For simplicity, this microcontroller, that is responsible for all the USB related operations, will be identifying the board as the Arduino Mega 2560 towards the Arduino IDE. This processor will be reprogrammable only via its SPI connector.
- The ATmega2560 is the one executing the operations commanded by the Linux core via serial communication. This board will come with a standard Arduino bootloader and will therefore be reprogrammable both through its SPI connector and the default serial port. If you used the SPI connector to reprogram the processor, the bootloader will be erased.

Both firmwares are open source and come as part of the official Arduino distribution. It is possible to reprogram them directly from the Arduino IDE. It is also possible to use other tools like command line, or AVR studio to change the firmware in both processors.

Open WRT

The Dogstamp block on the motherboard runs a distribution of Linux called Linino, that is based on OpenWrt³. To configure the system you have to access it from the command line, via a terminal through the serial port. There is the possibility of implementing a web page that would allow you to configure many of the different options available. The standard way of doing this, in OpenWRT is using the Lua⁴ programming language to build a web interface (called LuCI) that would give you access to almost any setting you would need for maintaining the WiFi interface.

To install additional software on Linino, you need to use the the Linux package manager *opkg*.

Further explanations about the reprogramming of the linux system are beyond the scope of this document. You should check the *Getting Started* guide for the board.

³ <http://openwrt.org>

⁴ <http://lua.org>

Licenses

Hardware

The design of the boards falls under the CERN Open Hardware License 1.2, for more information refer to the following link:

http://ohwr.org/attachments/2388/cern_ohl_v_1_2.txt

Firmware

The firmware for the ATmega16U2 and the ATmega2560 processors used in the SandS motherboard is based on the Arduino core and therefore it is licensed under LGPL, however all the source is available for anybody to use. To read more about this license, refer to:

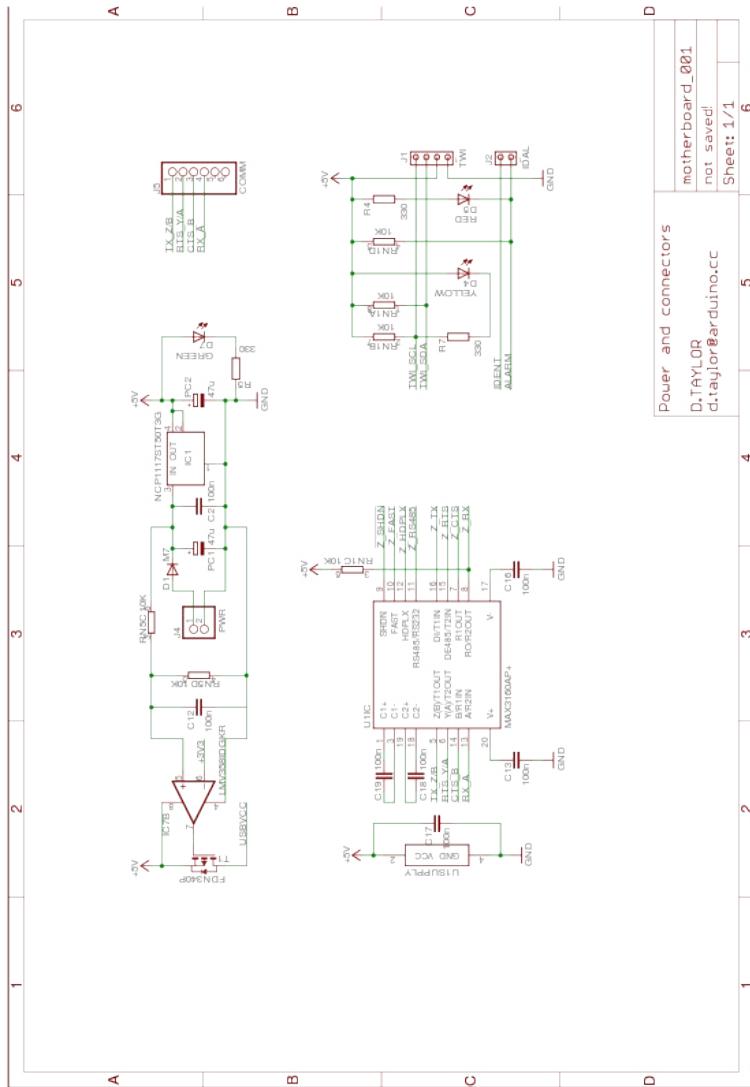
<https://www.gnu.org/licenses/lgpl.html>

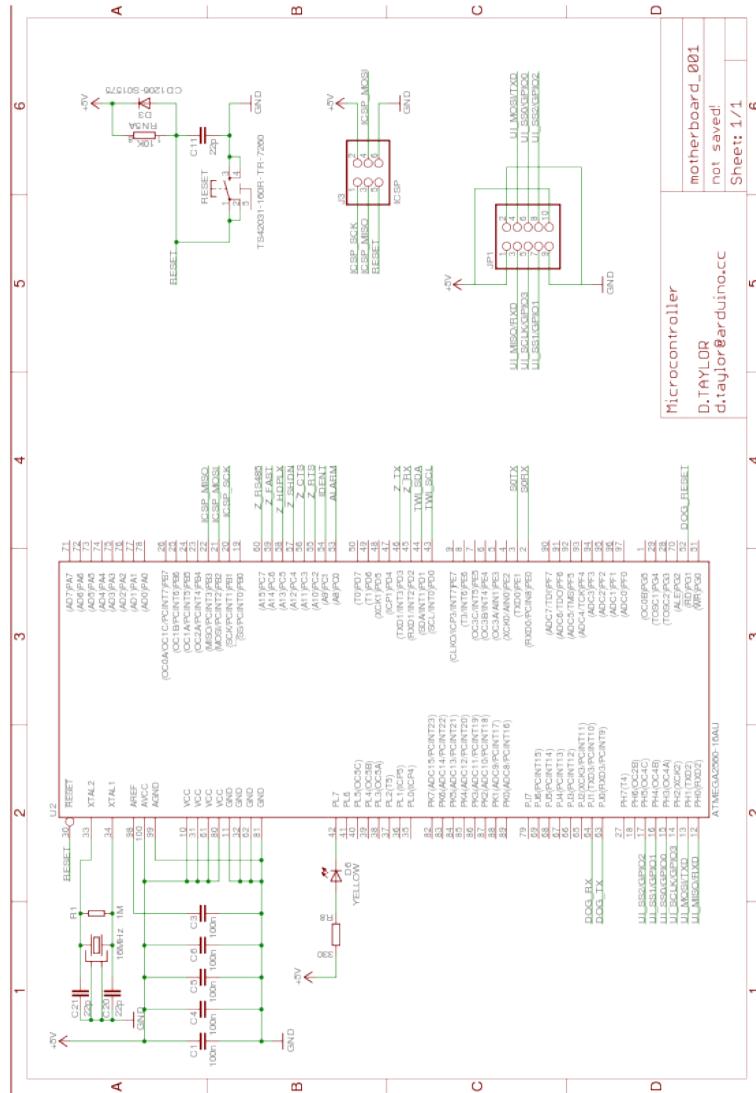
Linux distro

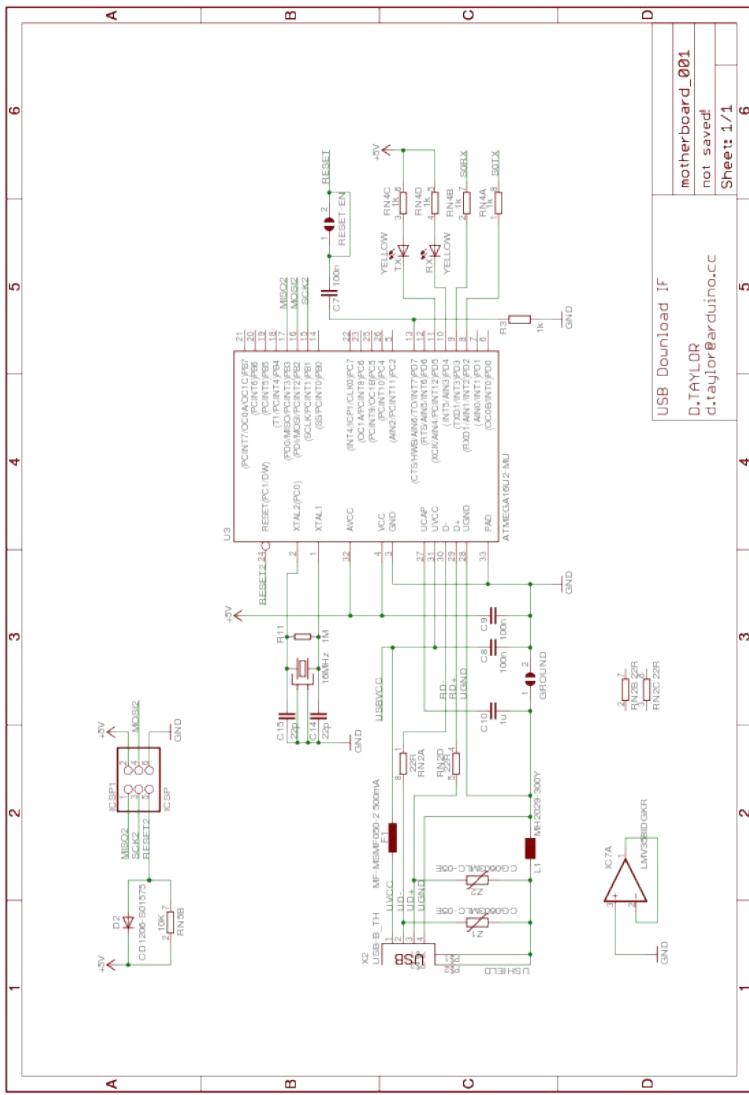
The Linino version of the OpenWRT used inside the Dogstamp is licensed under GPL v2, for more information visit:

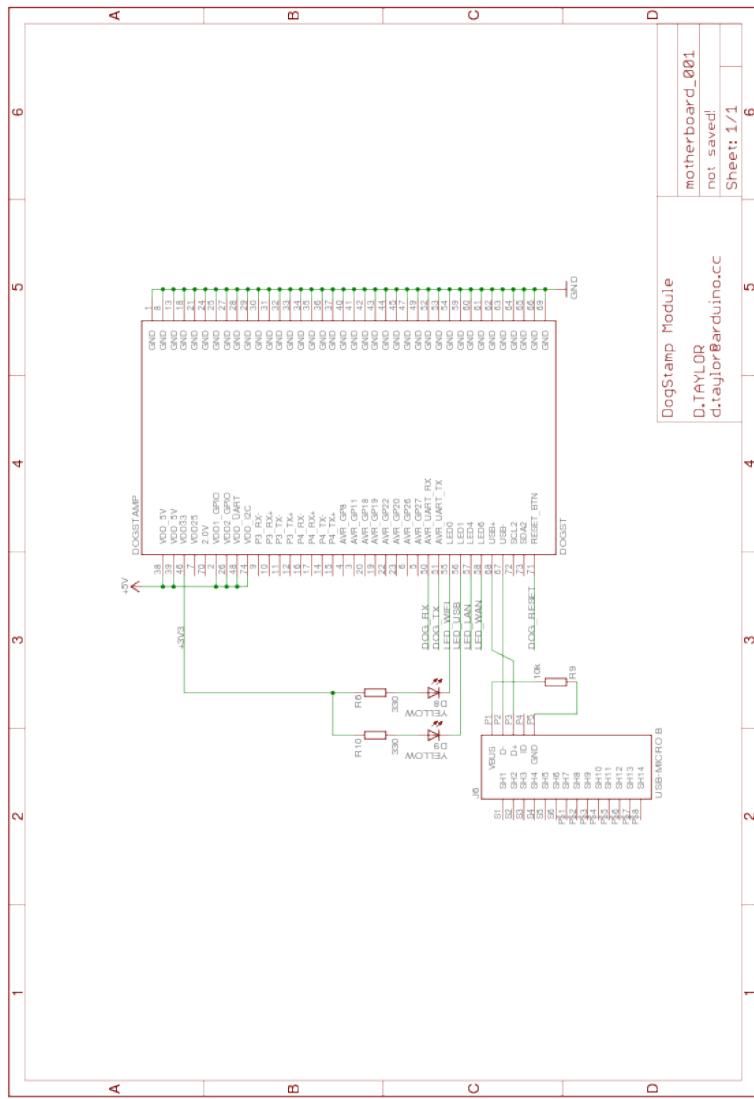
<http://www.gnu.org/licenses/gpl-2.0.html>

Appendix 1 - Schematics

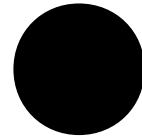








Document Information



Project reference

317947, FP7-ICT-2011-8

Context

Deliverable 2.1, workpackage 2

Document title

201309_Arduino_Sands_Deliverable_2-1_Consumption

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

1. Sep. 2013

Table of contents

Document Information.....	1
Table of contents.....	2
Summary.....	3
Purpose within SandS.....	3
SandS Consumption module.....	4
AVR core.....	4
Power the board.....	4
RMS-to-DC converter.....	4
Hardware.....	5
Board dimensions.....	6
Mounting considerations.....	6
Schematic.....	6
Board file.....	6
Pin layout.....	6
Firmware.....	6
Licenses.....	7
Hardware.....	7
Firmware.....	7
Appendix 1 – Schematics.....	8
Appendix 2 - Sensor Datasheet.....	10

Summary

This document describes the main characteristics of the SandS Consumption module, an open source hardware - Arduino compatible board, exposing a 5V serial port, and a SandS-enhanced I2C port. This board runs a series of commands to control its on-board power consumption sensor. In other words, it allows reading how much power is consumed by a certain appliance.

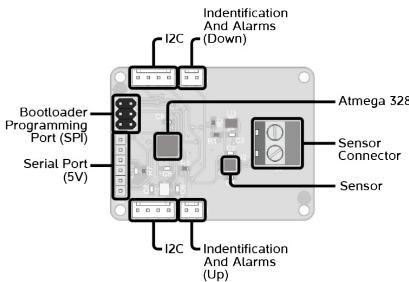


Image 1: block description, consumption module

The way this sensor works is by estimating the electromagnetic field produced by alternating current when crossing a conducting wire embraced by a coil. The coil is connected to the ATmega328 through a special IC.

Values are read with a 10-bit depth via the internal ADC peripheral inside the ATmega328.

Will read currents between 1 A and 50 A.

The board runs from the power provided by the I2C port. It can be reprogrammed from the Arduino IDE using a USB-Serial cable and the SandS special library. It comes with a pre-loaded firmware and runs a modified version of the I2C protocol that allows automatic identification. In other words, it is possible to dynamically assign addresses to the board.

Even if the board has been specifically designed for the SandS project, the pins exposed directly from the processor (the ATmega328) can be addressed as Arduino digital pins.

Purpose within SandS

The purpose of this module is to offer information about the power consumption of the appliance during different operations. The sensor chip can be hooked to different coils depending on the desired granularity of the measurement. Ideally the module should be part of every SandS appliance.

SandS Consumption module

The SandS consumption module is a microcontroller board operating as a I2C peripheral to the SandS motherboard. It is made of two blocks:

- a microcontroller (the ATmega328): running at 16MHz it listens to the I2C port and executes the different operations on the module
- an RMS to DC converter used to read the values measured by a split-core embracing a wire; those will be amplified and then read by the microprocessor

It comes with two connections for the SandS enhanced I2C (two-wire interface + IDENT + ALARM), one 5V RS-232 port, and the SPI programming port to burn the processor's bootloader. See Image 1 for a block diagram of the board.

AVR core

The AVR core chosen for the SandS consumption module is the ATmega328, which main characteristics are described in Table 1:

Processor	ATmega328
Flash Memory	32 KB of which 512 B used by bootloader
SRAM	2 KB
EEPROM	1 KB
Operating Voltage	5 V
Hardware Serial ports	1
Hardware SPI ports	1
Hardware I2C ports	1

Table 1: main characteristics of the AVR core

This processor can be programmed as an Arduino UNO board directly from the standard Arduino IDE, there is more information how to program the board on the *Getting Started* guide to the module.

The ATmega328 does not have an internal USB peripheral, therefore, the SandS consumption module needs using a USB-Serial cable, as a way to achieve USB communication.

Power the board

Input Voltage (recommended)	5 V through the I2C connector
Current (typ)	50mA

Table 2: powering the board

All the SandS modules are powered via the I2C cable. However, during reprogramming operations, it is possible to power the modules through the 5V serial connector.

RMS-to-DC converter

Current (max)	50 A
Current (min)	1 A
Voltage (max)	250V

Table 3: RMS-to-DC characteristics

The converter chosen for this circuit is the LTC1966 - Precision Micropower $\Delta\Sigma$ RMS-to-DC Converter from Linear instruments. It is a device that transforms and oscillating signal into a DC voltage, what makes it easy to read by microcontrollers. Readings are made by the ADC (Analog to Digital Converter) inside the ATmega328, which has 10 bits of depth.

For further information about the converter IC, it is recommended checking its datasheet, inside the Appendix 2.

Hardware

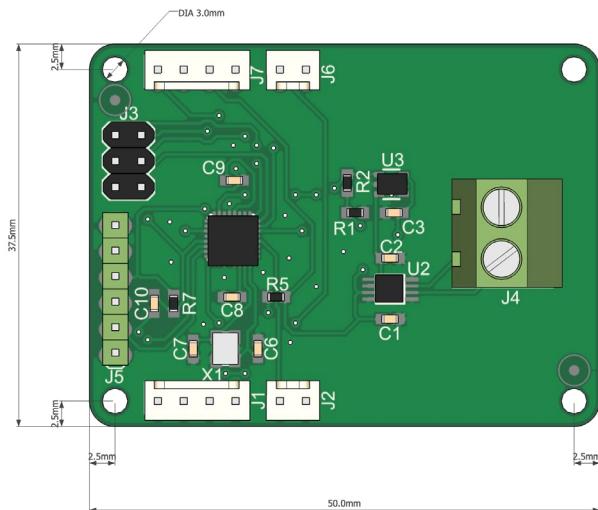


Image 2: Top view; SandS consumption module v0002

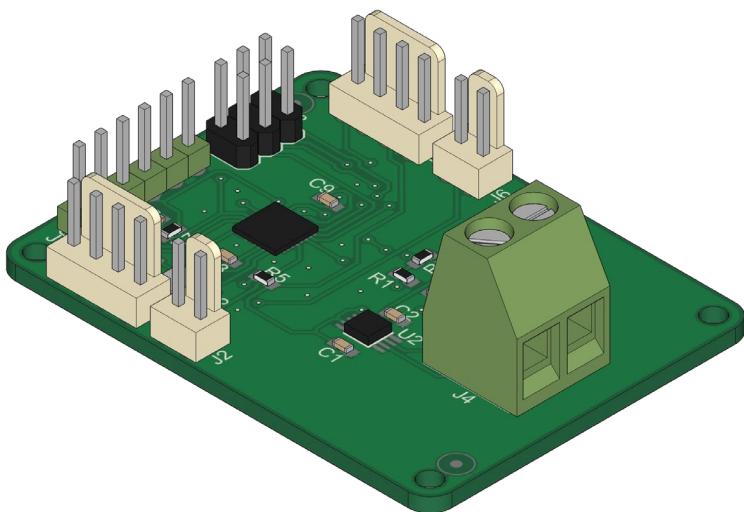


Image 3: Isometric view of the SandS consumption module v0002

Board dimensions

The board has been designed to easily accommodate the parts in a 2-layer design in the size of 37,5x50mm. It comes with 3mm holes at each corner, making it easy to mount it using standard plastic spacers and M3 screws.

Check Image 2 for further details about measurements and location of the holes on the board.

Image 3 gives you an idea of the proportions of the different components on the board.

Mounting considerations

It is recommended to mount the board in a plastic box.

The module will be connected as part of a I2C bus. This means that the board will use standard board-to-cable connectors both for the uplink and the downlink of the I2C and therefore it is recommended leaving space for those cables to exit the box. Alternatively it is possible to mount differen types of connectors on the plastic housing and simply extend the board with those.

The consumption module is meant to be reading the alternating signal flowing through a split-core embracing the power cord from an appliance. The sensing IC is accessible through a screw connector. If you were in the need to use a different type of connector, you should fix it to the board's box and plug it in to the screw connector.

Schematic

The board's schematic shows all the components on it. You can check all the schematics at Appendix 1, at the end of this document.

The schematics come as a single document showing:

- the RMS-to-DC converter
- an operational amplifiers used to amplify the signal coming from the converter
- Microcontroller: is the ATmega328, with a 16Mhz crystal
- the SPI connector for reprogramming the processor's bootloader
- the I2C uplink and downlink connectors
- the IDENT and ALARM connectors
- the serial port connection for reprogramming the device and debugging it's firmware

Board file

The board file is a 2-layered design with the components separated strategically to avoid potential interferences. Half of the board is reserved for the sensing circuitry,

while the other half is dedicated to the AVR core and all the connectors.

Images 2 and 3 give a pretty good idea of this layout. Both schematic and board file for this design come bundled in the same zipped file as this document and can be modified using the Eagle Cad software.

Pin layout

Image 4 shows the pin layout for the board as well as the description of each one of the pins separately.

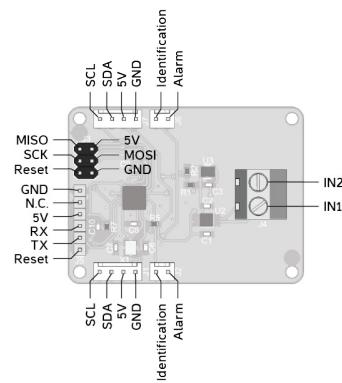


Image 4: pin layout, Consumption module

Firmware

The SandS consumption board comes with a preprogrammed firmware that will get the processor to receive commands through the I2C protocol and execute them accordingly. The normal operation mode doesn't need of reprogramming the device. All the specific commands for this device are described in depth in the TWI Protocol document that comes bundled with this one.

The firmware is open source and comes installed in the board. It is possible to reprogram it directly from the Arduino IDE. It is also possible to use other tools like command line, or AVR studio to change the firmware in the processor.

Licenses

Hardware

The design of the boards falls under the CERN Open Hardware License 1.2, for more information refer to the following link:

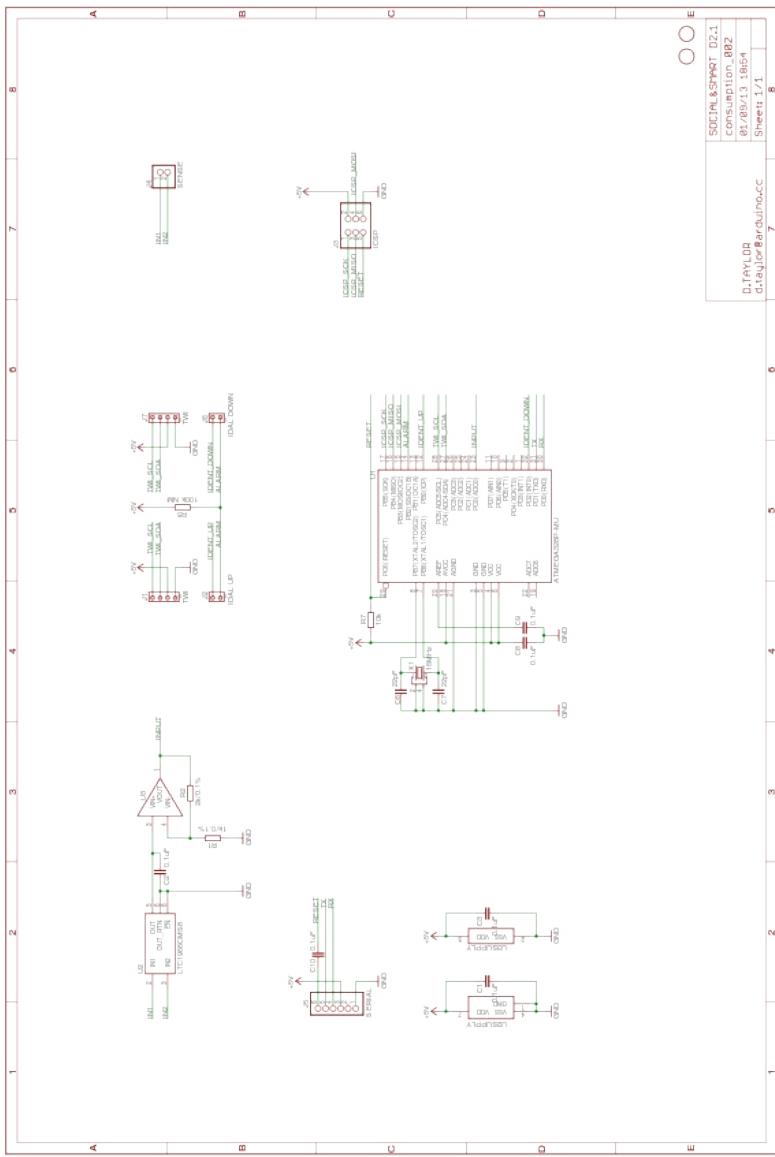
http://ohwr.org/attachments/2388/cern_ohl_v_1_2.txt

Firmware

The firmware for the ATmega328 processor used in the SandS Consumption board is based on the Arduino core and therefore it is licensed under LGPL, however all the source is available for anybody to use. To read more about this license, refer to:

<https://www.gnu.org/licenses/lgpl.html>

Appendix 1 – Schematics



Appendix 2 - Sensor Datasheet



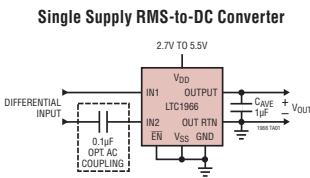
LTC1966

Precision Micropower
ΔΣ RMS-to-DC Converter

FEATURES

- Simple to Use, Requires One Capacitor
- True RMS DC Conversion Using ΔΣ Technology
- High Accuracy:
0.1% Gain Accuracy from 50Hz to 1kHz
0.25% Total Error from 50Hz to 1kHz
- High Linearity:
0.02% Linearity Allows Simple System Calibration
- Low Supply Current:
155µA Typ, 170µA Max
- Ultralow Shutdown Current:
0.1µA
- Constant Bandwidth:
Independent of Input Voltage
800kHz –3dB, 6kHz ±1%
- Flexible Supplies:
2.7V to 5.5V Single Supply
Up to ±5.5V Dual Supply
- Flexible Inputs:
Differential or Single-Ended
Rail-to-Rail Common Mode Voltage Range
Up to 1V_{PEAK} Differential Voltage
- Flexible Output:
Rail-to-Rail Output
Separate Output Reference Pin Allows Level Shifting
- Wide Temperature Range:
–55°C to 125°C
- Small Size:
Space Saving 8-Pin MSOP Package

TYPICAL APPLICATION



DESCRIPTION

The LTC®1966 is a true RMS-to-DC converter that utilizes an innovative patented ΔΣ computational technique. The internal delta sigma circuitry of the LTC1966 makes it simpler to use, more accurate, lower power and dramatically more flexible than conventional log antilog RMS-to-DC converters.

The LTC1966 accepts single-ended or differential input signals (for EMI/RFI rejection) and supports crest factors up to 4. Common mode input range is rail-to-rail. Differential input range is 1V_{PEAK}, and offers unprecedented linearity. Unlike previously available RMS-to-DC converters, the superior linearity of the LTC1966 allows hassle free system calibration at any input voltage.

The LTC1966 also has a rail-to-rail output with a separate output reference pin providing flexible level shifting. The LTC1966 operates on a single power supply from 2.7V to 5.5V or dual supplies up to ±5.5V. A low power shutdown mode reduces supply current to 0.5µA.

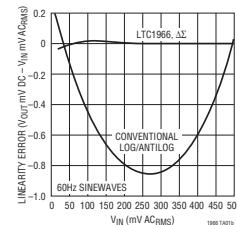
The LTC1966 is insensitive to PC board soldering and stresses, as well as operating temperature. The LTC1966 is packaged in the space saving MSOP package which is ideal for portable applications.

APPLICATIONS

- True RMS Digital Multimeters and Panel Meters
- True RMS AC + DC Measurements

LT, LT, LTC, LTM, Linear Technology and the Linear logo are registered trademarks and No Latency ΔΣ is a trademark of Linear Technology Corporation. All other trademarks are the property of their respective owners. Protected by U.S. Patents including 6359576, 6362677, 6516291 and 6651036.

Quantum Leap in Linearity Performance



1966fb



LTC1966

ABSOLUTE MAXIMUM RATINGS

(Note 1)

Supply Voltage

V _{DD} to GND	-0.3V to 7V
V _{DD} to V _{SS}	-0.3V to 12V
V _{SS} to GND	-7V to 0.3V

Input Currents (Note 2)

±10mA

Output Current (Note 3)

±10mA

ENABLE Voltage

V_{SS} - 0.3V to V_{SS} + 12V

OUT RTN Voltage

V_{SS} - 0.3V to V_{DD}

Operating Temperature Range (Note 4)

LTC1966C/LTC1966I

-40°C to 85°C

LTC1966H

-40°C to 125°C

LTC1966MP

-55°C to 125°C

Specified Temperature Range (Note 5)

LTC1966C/LTC1966I

-40°C to 85°C

LTC1966H

-40°C to 125°C

LTC1966MP

-55°C to 125°C

Maximum Junction Temperature

150°C

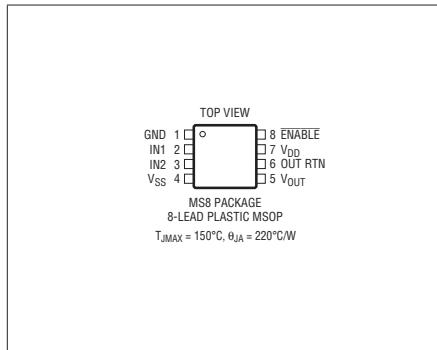
Storage Temperature Range

-65°C to 150°C

Lead Temperature (Soldering, 10 sec)

300°C

PIN CONFIGURATION



MS8 PACKAGE
8-LEAD PLASTIC MSOP

T_{JMAX} = 150°C, θ_{JA} = 220°C/W

ORDER INFORMATION

LEAD FREE FINISH	TAPE AND REEL	PART MARKING*	PACKAGE DESCRIPTION	TEMPERATURE RANGE
LTC1966CMS8#PBF	LTC1966CMS8#TRPB	LTTG	8-Lead Plastic MSOP	0°C to 70°C
LTC1966IMS8#PBF	LTC1966IMS8#TRPB	LTHH	8-Lead Plastic MSOP	-40°C to 85°C
LTC1966HMS8#PBF	LTC1966HMS8#TRPB	LTTG	8-Lead Plastic MSOP	-40°C to 125°C
LTC1966MPMS8#PBF	LTC1966MPMS8#TRPB	LTTG	8-Lead Plastic MSOP	-55°C to 125°C

Consult LTC Marketing for parts specified with wider operating temperature ranges. *The temperature grade is identified by a label on the shipping container.

For more information on lead free part marking, go to: <http://www.linear.com/leadfree/>

For more information on tape and reel specifications, go to: <http://www.linear.com/tapeandree/>

ELECTRICAL CHARACTERISTICS The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at T_A = 25°C. V_{DD} = 5V, V_{SS} = -5V, V_{OUTRTN} = 0V, C_{AVE} = 10μF, V_{IN} = 200mV_{RMS}, V_{ENABLE} = 0.5V unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Conversion Accuracy						
G _{ERR}	Conversion Gain Error	50Hz to 1kHz Input (Notes 6, 7) LTC1966C, LTC1966I LTC1966H, LTC1966MP	●	±0.1	±0.3	%
●			●	±0.4	±0.7	%
V _{OOS}	Output Offset Voltage	(Notes 6, 7) LTC1966C, LTC1966I LTC1966H, LTC1966MP	●	0.1	0.2	mV
●			●	0.4	0.6	mV
LIN _{ERR}	Linearity Error	50mV to 350mV (Notes 7, 8)	●	0.02	0.15	%

1966fb

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at $T_A = 25^\circ\text{C}$. $V_{DD} = 5\text{V}$, $V_{SS} = -5\text{V}$, $V_{OUTRTN} = 0\text{V}$, $C_{AVE} = 10\mu\text{F}$, $V_{IN} = 200\text{mV}_{\text{RMS}}$, $V_{ENABLE} = 0.5\text{V}$ unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
PSRR	Power Supply Rejection	(Note 9) LTC1966C, LTC1966I LTC1966H, LTC1966MP	● ●	0.02 0.20 0.3	0.15 0.20 0.3	%/V %/V %/V
V_{IOS}	Input Offset Voltage	(Notes 6, 7, 10)	●	0.02	0.8 1.0	mV mV
Accuracy vs Crest Factor (CF)						
	CF = 4	60Hz Fundamental, 200mV _{RMS} (Note 11)	●	-1	2	mV
	CF = 5	60Hz Fundamental, 200mV _{RMS} (Note 11)	●	-20	30	mV
Input Characteristics						
I_{VR}	Input Voltage Range	(Note 14)	●	V_{SS}	V_{DD}	V
Z_{IN}	Input Impedance	Average, Differential (Note 12) Average, Common Mode (Note 12)		8 100		MΩ MΩ
CMRR _I	Input Common Mode Rejection	(Note 13)	●	7	200	μV/V
V_{IMAX}	Maximum Input Swing	Accuracy = 1% (Note 14)	●	1	1.05	V
V_{IMIN}	Minimum RMS Input		●		5	mV
PSRR _I	Power Supply Rejection	V_{DD} Supply (Note 9) V_{SS} Supply (Note 9)	● ●	250 120	600 300	μV/V μV/V
Output Characteristics						
OVR	Output Voltage Range		●	V_{SS}	V_{DD}	V
Z_{OUT}	Output Impedance	$V_{ENABLE} = 0.5\text{V}$ (Note 12) $V_{ENABLE} = 4.5\text{V}$	●	75 30	95	kΩ kΩ
CMRRO	Output Common Mode Rejection	(Note 13)	●	16	200	μV/V
V_{OMAX}	Maximum Differential Output Swing	Accuracy = 2%, DC Input (Note 14)	●	1.0 0.9	1.05	V V
PSRRO	Power Supply Rejection	V_{DD} Supply (Note 9) V_{SS} Supply (Note 9)	● ●	250 50	1000 500	μV/V μV/V
Frequency Response						
f_{1P}	1% Additional Error (Note 15)	$C_{AVE} = 10\mu\text{F}$			6	kHz
f_{10P}	10% Additional Error (Note 15)	$C_{AVE} = 10\mu\text{F}$			20	kHz
f_{-3dB}	±3dB Frequency (Note 15)				800	kHz
Power Supplies						
V_{DD}	Positive Supply Voltage		●	2.7	5.5	V
V_{SS}	Negative Supply Voltage	(Note 16)	●	-5.5	0	V
I_{DD}	Positive Supply Current	$IN_1 = 20\text{mV}, IN_2 = 0\text{V}$ $IN_1 = 200\text{mV}, IN_2 = 0\text{V}$	●	155 158	170	μA μA
I_{SS}	Negative Supply Current	$IN_1 = 20\text{mV}, IN_2 = 0\text{V}$	●	12	20	μA
Shutdown Characteristics						
I_{DDS}	Supply Currents	$V_{ENABLE} = 4.5\text{V}$	●	0.5	10	μA
I_{SSS}	Supply Currents	$V_{ENABLE} = 4.5\text{V}$ LTC1966H, LTC1966MP	● ●	-1 -2	-0.1	μA μA
I_{IH}	ENABLE Pin Current High	$V_{ENABLE} = 4.5\text{V}$	●	-0.3	-0.05	μA

1966fb

LTC1966

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at $T_A = 25^\circ\text{C}$. $V_{DD} = 5\text{V}$, $V_{SS} = -5\text{V}$, $V_{OUTRTN} = 0\text{V}$, $C_{AVE} = 10\text{pF}$, $V_{IN} = 200\text{mVRMS}$, $V_{ENABLE} = 0.5\text{V}$ unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
I_{IL}	ENABLE Pin Current Low	$V_{ENABLE} = 0.5\text{V}$ $\text{LTC1966H, LTC1966MP}$	●	-2	-1	-0.1	μA
V_{TH}	ENABLE Threshold Voltage	$V_{DD} = 5\text{V}, V_{SS} = -5\text{V}$ $V_{DD} = 5\text{V}, V_{SS} = \text{GND}$ $V_{DD} = 2.7\text{V}, V_{SS} = \text{GND}$			2.4		V
V_{HYS}	ENABLE Threshold Hysteresis				0.1		V

Note 1: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. Exposure to any Absolute Maximum Rating condition for extended periods may affect device reliability and lifetime.

Note 2: The inputs ($\text{IN}1$, $\text{IN}2$) are protected by shunt diodes to V_{SS} and V_{DD} . If the inputs are driven beyond the rails, the current should be limited to less than 10mA.

Note 3: The LTC1966 output (V_{out}) is high impedance and can be overdriven, either sinking or sourcing current, to the limits stated.

Note 4: The LTC1966C/LTC1966I are guaranteed functional over the operating temperature range of -40°C to 85°C . The LTC1966H/LTC1966MP are guaranteed functional over the operating temperature range of -55°C to 125°C .

Note 5: The LTC1966C is guaranteed to meet specified performance from 0°C to 70°C . The LTC1966C is designed, characterized and expected to meet specified performance from -40°C to 85°C but is not tested nor QA sampled at these temperatures. The LTC1966I is guaranteed to meet specified performance from -40°C to 85°C . The LTC1966H is guaranteed to meet specified performance from -40°C to 125°C . The LTC1966MP is guaranteed to meet specified performance from -55°C to 125°C .

Note 6: High speed automatic testing cannot be performed with $C_{AVE} = 10\text{pF}$. The LTC1966 is 100% tested with $C_{AVE} = 22\text{nF}$. Correlation tests have shown that the performance limits above can be guaranteed with the additional testing being performed to guarantee proper operation of all the internal circuitry.

Note 7: High speed automatic testing cannot be performed with 60Hz inputs. The LTC1966 is 100% tested with DC and 10kHz input signals. Measurements from DC inputs from 50mV to 350mV are used to calculate the four parameters: G_{ERR} , V_{DOS} , V_{IQS} and linearity error. Correlation tests have shown that the performance limits above can be guaranteed with the additional testing being performed to guarantee proper operation of all internal circuitry.

Note 8: The LTC1966 is inherently very linear. Unlike older log/antilog circuits, its behavior is the same with DC and AC inputs, and DC inputs are used for high speed testing.

Note 9: The power supply rejections of the LTC1966 are measured with DC inputs from 50mV to 350mV. The change in accuracy from $V_{DD} = 2.7\text{V}$ to $V_{DD} = 5.5\text{V}$ with $V_{SS} = 0\text{V}$ is divided by 2.8V. The change in accuracy from $V_{SS} = 0\text{V}$ to $V_{SS} = -5.5\text{V}$ with $V_{DD} = 5.5\text{V}$ is divided by 5.5V.

Note 10: Previous generation RMS-to-DC converters required nonlinear input stages as well as a nonlinear core. Some parts specify a DC reversal error, combining the effects of input nonlinearity and input offset voltage. The LTC1966 behavior is simpler to characterize and the input offset voltage is the only significant source of DC reversal error.

Note 11: High speed automatic testing cannot be performed with 60Hz inputs. The LTC1966 is 100% tested with DC stimulus. Correlation tests have shown that the performance limits above can be guaranteed with the additional testing being performed to verify proper operation of all internal circuitry.

Note 12: The LTC1966 is a switched capacitor device and the input/output impedance is an average impedance over many clock cycles. The input impedance will not necessarily lead to an attenuation of the input signal measured. Refer to the Applications Information section titled Input Impedance for more information.

Note 13: The common mode rejection ratios of the LTC1966 are measured with DC inputs from 50mV to 350mV. The input CMRR is defined as the change in V_{IQS} measured between input levels of $V_{SS} + 350\text{mV}$ and input levels of $V_{DD} - 350\text{mV}$ to V_{DD} divided by $V_{DD} - V_{SS} - 350\text{mV}$. The output CMRR is defined as the change in V_{OQS} measured with OUT RTN = V_{SS} and OUT RTN = $V_{DD} - 350\text{mV}$ divided by $V_{DD} - V_{SS} - 350\text{mV}$.

Note 14: Each input of the LTC1966 can withstand any voltage within the supply range. These inputs are protected with ESD diodes, so going beyond the supply voltages can damage the part if the absolute maximum current ratings are exceeded. Likewise for the output pins. The LTC1966 input and output voltage swings are limited by internal clipping. The maximum differential input of the LTC1966 (referred to as maximum input swing) is 1V. This applies to either input polarity, so it can be thought of as $\pm 1\text{V}$. Because the differential input voltage gets processed by the LTC1966 with gain, it is subject to internal clipping. Exceeding the 1V maximum can, depending on the input crest factor, impact the accuracy of the output voltage, but does not damage the part. Fortunately, the LTC1966's $\Delta\Sigma$ topology is relatively tolerant of momentary internal clipping. The input clipping is tested with a crest factor of 2, while the output clipping is tested with a DC input.

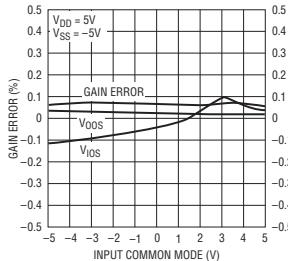
Note 15: The LTC1966 exploits oversampling and noise shaping to reduce the quantization noise of internal 1-bit analog-to-digital conversions. At higher input frequencies, increasingly large portions of this noise are aliased down to DC. Because the noise is shifted in frequency, it becomes a low frequency rumble and is only filtered at the expense of increasingly long settling times. The LTC1966 is inherently wideband, but the output accuracy is degraded by this aliased noise. These specifications apply with $C_{AVE} = 10\text{pF}$ and constitute a 3-sigma variation of the output rumble.

Note 16: The LTC1966 can operate down to 2.7V single supply but cannot operate at $\pm 2.7\text{V}$. This additional constraint on V_{SS} can be expressed mathematically as $-3 \cdot (V_{DD} - 2.7\text{V}) \leq V_{SS} \leq \text{Ground}$.

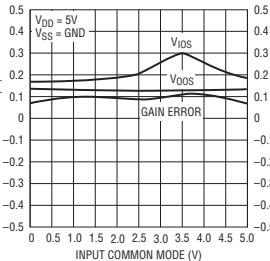
1966fb

TYPICAL PERFORMANCE CHARACTERISTICS

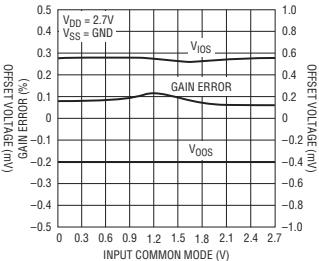
Gain and Offsets vs Input Common Mode



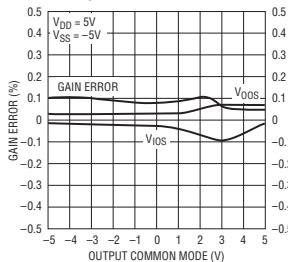
Gain and Offsets vs Input Common Mode



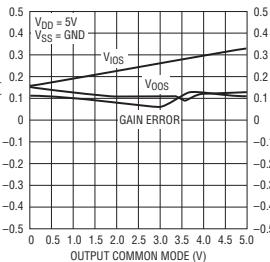
Gain and Offsets vs Input Common Mode



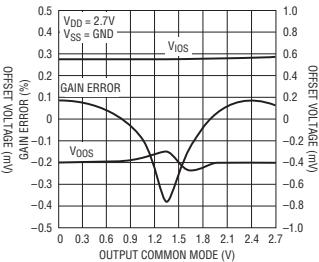
Gain and Offsets vs Output Common Mode



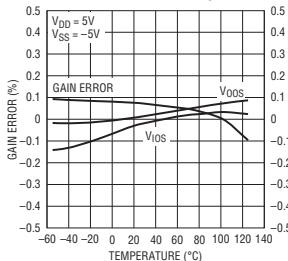
Gain and Offsets vs Output Common Mode



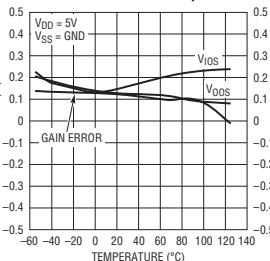
Gain and Offsets vs Output Common Mode



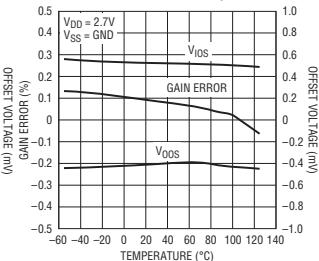
Gain and Offsets vs Temperature



Gain and Offsets vs Temperature



Gain and Offsets vs Temperature

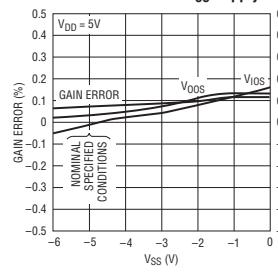


1966fb

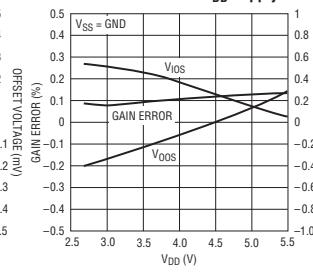
LTC1966

TYPICAL PERFORMANCE CHARACTERISTICS

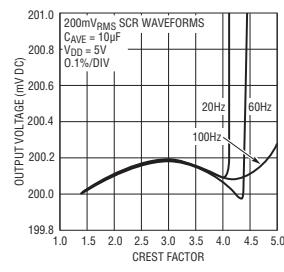
Gain and Offsets vs V_{SS} Supply



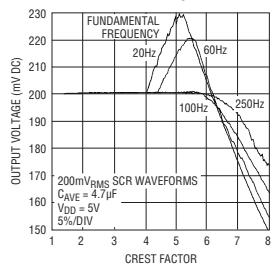
Gain and Offsets vs V_{DD} Supply



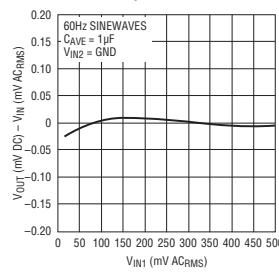
Performance vs Crest Factor



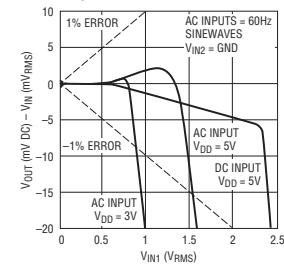
Performance vs Large Crest Factors



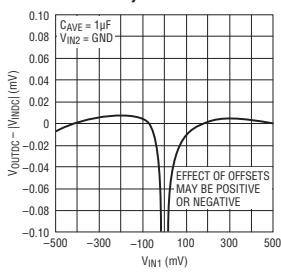
AC Linearity



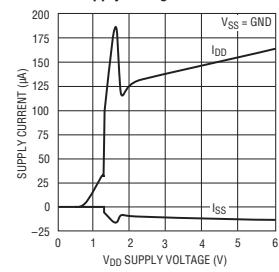
Output Accuracy vs Signal Amplitude



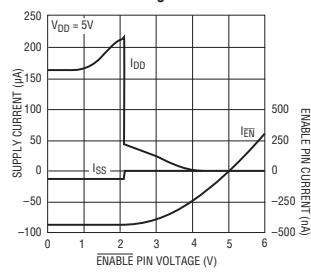
DC Linearity



Quiescent Supply Currents vs Supply Voltage

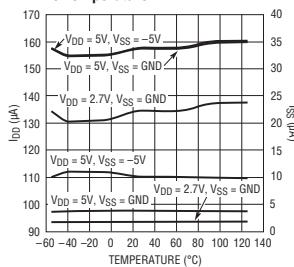


Shutdown Currents vs ENABLE Voltage

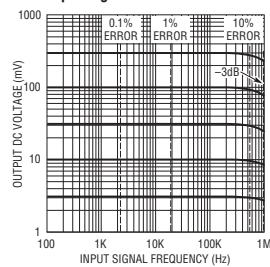


TYPICAL PERFORMANCE CHARACTERISTICS

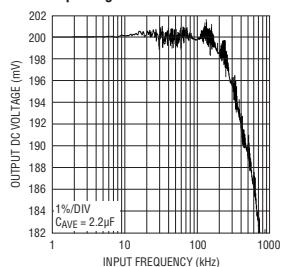
Quiescent Supply Currents vs Temperature



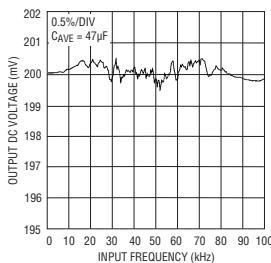
Input Signal Bandwidth



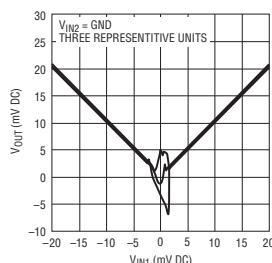
Input Signal Bandwidth



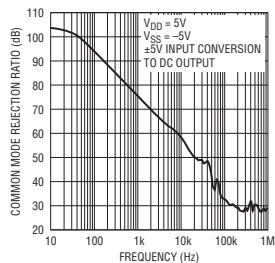
Bandwidth to 100kHz



DC Transfer Function Near Zero



Common Mode Rejection Ratio vs Frequency



LTC1966

PIN FUNCTIONS

GND (Pin 1): Ground. A power return pin.

IN1 (Pin 2): Differential Input. DC coupled (polarity is irrelevant).

IN2 (Pin 3): Differential Input. DC coupled (polarity is irrelevant).

V_{SS} (Pin 4): Negative Voltage Supply. GND to -5.5V.

V_{OUT} (Pin 5): Output Voltage. This is high impedance. The RMS averaging is accomplished with a single shunt capacitor from this node to OUT RTN. The transfer function is given by:

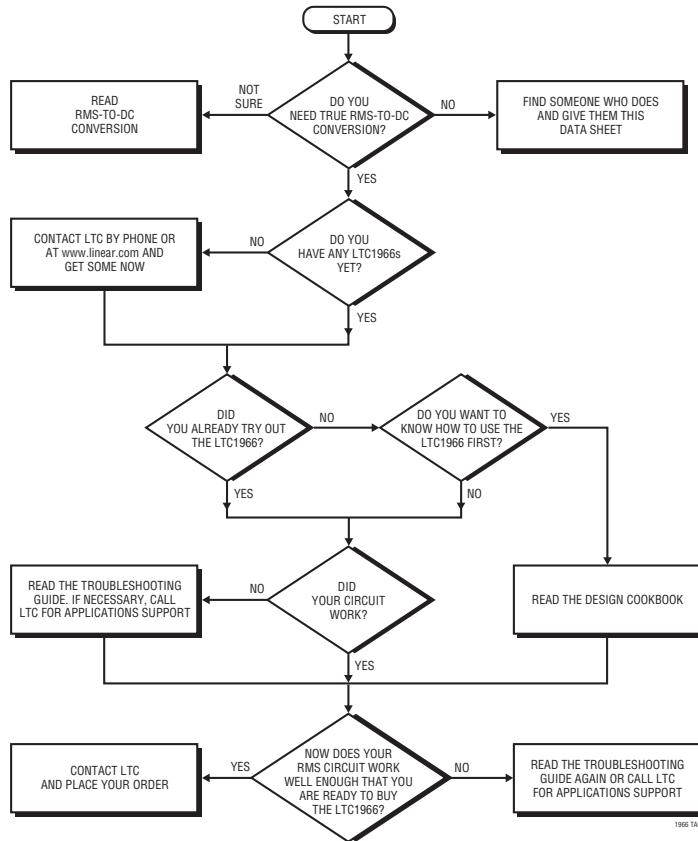
$$(V_{OUT} - \text{OUT RTN}) = \sqrt{\text{Average}[(IN2 - IN1)^2]}$$

OUT RTN (Pin 6): Output Return. The output voltage is created relative to this pin. The V_{OUT} and OUT RTN pins are not balanced and this pin should be tied to a low impedance, both AC and DC. Although it is typically tied to GND, it can be tied to any arbitrary voltage, V_{SS} < OUT RTN < (V_{DD} - Max Output). Best results are obtained when OUT RTN = GND.

V_{DD} (Pin 7): Positive Voltage Supply. 2.7V to 5.5V.

ENABLE (Pin 8): An Active Low Enable Input. LTC1966 is debiased if open circuited or driven to V_{DD}. For normal operation, pull to GND, a logic low or even V_{SS}.

APPLICATIONS INFORMATION



1966 TA02

APPLICATIONS INFORMATION

RMS-TO-DC CONVERSION

Definition of RMS

RMS amplitude is the consistent, fair and standard way to measure and compare dynamic signals of all shapes and sizes. Simply stated, the RMS amplitude is the heating potential of a dynamic waveform. A 1V_{RMS} AC waveform will generate the same heat in a resistive load as will 1V DC.

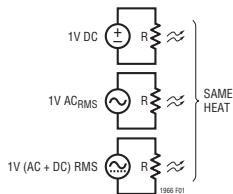


Figure 1

Mathematically, RMS is the root of the mean of the square:

$$V_{RMS} = \sqrt{V^2}$$

Alternatives to RMS

Other ways to quantify dynamic waveforms include peak detection and average rectification. In both cases, an average (DC) value results, but the value is only accurate at the one chosen waveform type for which it is calibrated, typically sine waves. The errors with average rectification are shown in Table 1. Peak detection is worse in all cases and is rarely used.

Table 1. Errors with Average Rectification vs True RMS

WAVEFORM	V_{RMS}	AVERAGE RECTIFIED (V)	ERROR*
Square Wave	1.000	1.000	11%
Sine Wave	1.000	0.900	*Calibrate for 0% Error
Triangle Wave	1.000	0.866	-3.8%
SCR at 1/2 Power, $\Theta = 90^\circ$	1.000	0.637	-29.3%
SCR at 1/4 Power, $\Theta = 114^\circ$	1.000	0.536	-40.4%

The last two entries of Table 1 are chopped sine waves as is commonly created with thyristors such as SCRs and Triacs. Figure 2a shows a typical circuit and Figure 2b shows the resulting load voltage, switch voltage and load currents. The power delivered to the load depends on the firing angle, as well as any parasitic losses such as switch ON voltage drop. Real circuit waveforms will also typically have significant ringing at the switching transition, dependent on exact circuit parasitics. For the purposes of this data sheet, SCR waveforms refers to the ideal chopped sine wave, though the LTC1966 will do faithful RMS-to-DC conversion with real SCR waveforms as well.

The case shown is for $\Theta = 90^\circ$, which corresponds to 50% of available power being delivered to the load. As noted in Table 1, when $\Theta = 114^\circ$, only 25% of the available power is being delivered to the load and the power drops quickly as Θ approaches 180°.

With an average rectification scheme and the typical calibration to compensate for errors with sine waves, the RMS level of an input sine wave is properly reported; it is only with a nonsinusoidal waveform that errors occur. Because of this calibration, and the output reading in V_{RMS} , the term true RMS got coined to denote the use of an actual RMS-to-DC converter as opposed to a calibrated average rectifier.

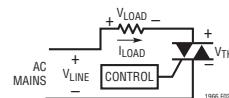


Figure 2a

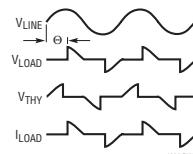


Figure 2b

APPLICATIONS INFORMATION

How an RMS-to-DC Converter Works

Monolithic RMS-to-DC converters use an implicit computation to calculate the RMS value of an input signal. The fundamental building block is an analog multiply/divide used as shown in Figure 3. Analysis of this topology is easy and starts by identifying the inputs and the output of the lowpass filter. The input to the LPF is the calculation from the multiplier/divider; $(V_{IN})^2/V_{OUT}$. The lowpass filter will take the average of this to create the output, mathematically:

$$V_{OUT} = \sqrt{\frac{(V_{IN})^2}{V_{OUT}}}$$

Because V_{OUT} is DC,

$$\sqrt{\frac{(V_{IN})^2}{V_{OUT}}} = \sqrt{\frac{(V_{IN})^2}{V_{OUT}}}, \text{ so}$$

$$V_{OUT} = \frac{(V_{IN})^2}{V_{OUT}}, \text{ and}$$

$$(V_{OUT})^2 = (V_{IN})^2, \text{ or}$$

$$V_{OUT} = \sqrt{(V_{IN})^2} = \text{RMS}(V_{IN})$$

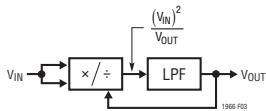


Figure 3. RMS-to-DC Converter with Implicit Computation

Unlike the prior generation RMS-to-DC converters, the LTC1966 computation does NOT use log/antilog circuits, which have all the same problems, and more, of log/antilog multipliers/dividers, i.e., linearity is poor, the bandwidth changes with the signal amplitude and the gain drifts with temperature.

How the LTC1966 RMS-to-DC Converter Works

The LTC1966 uses a completely new topology for RMS-to-DC conversion, in which a $\Delta\Sigma$ modulator acts as the divider, and a simple polarity switch is used as the multiplier as shown in Figure 4.

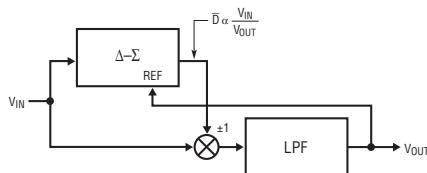


Figure 4. Topology of LTC1966

The $\Delta\Sigma$ modulator has a single-bit output whose average duty cycle (\bar{D}) will be proportional to the ratio of the input signal divided by the output. The $\Delta\Sigma$ is a 2nd order modulator with excellent linearity. The single bit output is used to selectively buffer or invert the input signal. Again, this is a circuit with excellent linearity, because it operates at only two points: ± 1 gain; the average effective multiplication over time will be on the straight line between these two points. The combination of these two elements again creates a lowpass filter input signal proportional to $(V_{IN})^2/V_{OUT}$, which, as shown above, results in RMS-to-DC conversion.

The lowpass filter performs the averaging of the RMS function and must be a lower corner frequency than the lowest frequency of interest. For line frequency measurements, this filter is simply too large to implement on-chip, but the LTC1966 needs only one capacitor on the output to implement the lowpass filter. The user can select this capacitor depending on frequency range and settling time requirements, as will be covered in the Design Cookbook section to follow.

This topology is inherently more stable and linear than log/antilog implementations primarily because all of the signal processing occurs in circuits with high gain op amps operating closed loop.

LTC1966

APPLICATIONS INFORMATION

More detail of the LTC1966 inner workings is shown in the Simplified Schematic towards the end of this data sheet. Note that the internal scalings are such that the $\Delta\Sigma$ output duty cycle is limited to 0% or 100% only when V_{IN} exceeds $\pm 4 \cdot V_{OUT}$.

Linearity of an RMS-to-DC Converter

Linearity may seem like an odd property for a device that implements a function that includes two very nonlinear processes: squaring and square rooting.

However, an RMS-to-DC converter has a transfer function, RMS volts in to DC volts out, that should ideally have a 1:1 transfer function. To the extent that the input to output transfer function does not lie on a straight line, the part is nonlinear.

A more complete look at linearity uses the simple model shown in Figure 5. Here an ideal RMS core is corrupted by both input circuitry and output circuitry that have imperfect transfer functions. As noted, input offset is introduced in the input circuitry, while output offset is introduced in the output circuitry.

Any nonlinearity that occurs in the output circuitry will corrupt the RMS in to DC out transfer function. A nonlinearity in the input circuitry will typically corrupt that transfer function far less, simply because with an AC input, the RMS-to-DC conversion will average the nonlinearity from a whole range of input values together.

But the input nonlinearity will still cause problems in an RMS-to-DC converter because it will corrupt the accuracy as the input signal shape changes. Although an RMS-to-DC converter will convert any input waveform to a DC output, the accuracy is not necessarily as good for all waveforms as it is with sine waves. A common way to describe dynamic signal wave shapes is crest factor. The crest factor is the ratio of the peak value relative to the RMS value of a waveform. A signal with a crest factor of 4, for instance, has a peak that is four times its RMS value. Because this peak has energy (proportional to voltage squared) that is 16 times (4^2) the energy of the RMS value, the peak is necessarily present for at most 6.25% (1/16) of the time.

The LTC1966 performs very well with crest factors of 4 or less and will respond with reduced accuracy to signals with higher crest factors. The high performance with crest factors less than 4 is directly attributable to the high linearity throughout the LTC1966.

The LTC1966 does not require an input rectifier, as is common with traditional log/antilog RMS-to-DC converters. Thus, the LTC1966 has none of the nonlinearities that are introduced by rectification.

The excellent linearity of the LTC1966 allows calibration to be highly effective at reducing system errors. See System Calibration section following the Design Cookbook.



Figure 5. Linearity Model of an RMS-to-DC Converter

APPLICATIONS INFORMATION

DESIGN COOKBOOK

The LTC1966 RMS-to-DC converter makes it easy to implement a rather quirky function. For many applications all that will be needed is a single capacitor for averaging, appropriate selection of the I/O connections and power supply bypassing. Of course, the LTC1966 also requires power. A wide variety of power supply configurations are shown in the Typical Applications section towards the end of this data sheet.

Capacitor Value Selection

The RMS or root-mean-squared value of a signal, *the root of the mean of the square*, cannot be computed without some averaging to obtain the *mean* function. The LTC1966 true RMS-to-DC converter utilizes a single capacitor on the output to do the low frequency averaging required for RMS-to-DC conversion. To give an accurate measure of a dynamic waveform, the averaging must take place over a sufficiently long interval to average, rather than track, the lowest frequency signals of interest. For a single averaging capacitor, the accuracy at low frequencies is depicted in Figure 6.

Figure 6 depicts the so-called DC error that results at a given combination of input frequency and filter capacitor values¹. It is appropriate for most applications, in which the output is fed to a circuit with an inherently band limited frequency response, such as a dual slope/integrating A/D converter, a Δ A/D converter or even a mechanical analog meter.

However, if the output is examined on an oscilloscope with a very low frequency input, the incomplete averaging will be seen, and this ripple will be larger than the error depicted in Figure 6. Such an output is depicted in Figure 7. The ripple is at twice the frequency of the input because of the computation of the square of the input. The typical values shown, 5% peak ripple with 0.05% DC error, occur with $C_{AVE} = 1\mu F$ and $f_{INPUT} = 10Hz$.

If the application calls for the output of the LTC1966 to feed a sampling or Nyquist A/D converter (or other circuitry that will not average out this double frequency ripple) a larger averaging capacitor can be used. This trade-off is depicted in Figure 8. The peak ripple error can also be reduced by additional lowpass filtering after the LTC1966, but the simplest solution is to use a larger averaging capacitor.

¹This frequency dependent error is in addition to the static errors that affect all readings and are therefore easy to trim or calibrate out. The Error Analyses section to follow discusses the effect of static error terms.

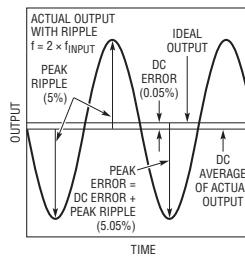


Figure 7. Output Ripple Exceeds DC Error

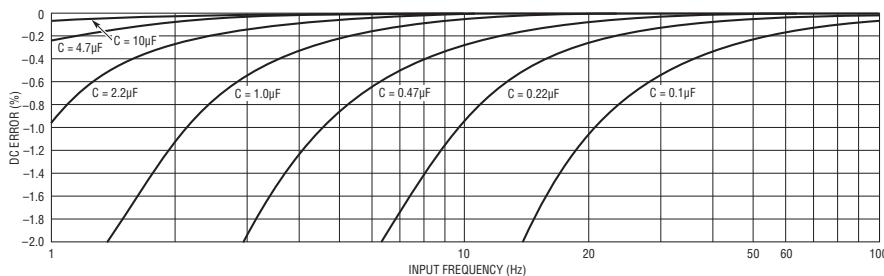


Figure 6. DC Error vs Input Frequency

1966FB

APPLICATIONS INFORMATION

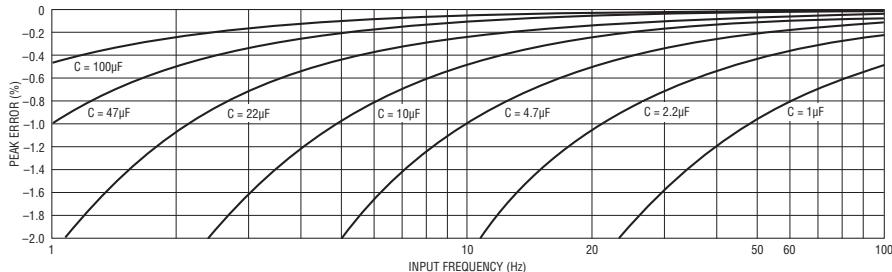


Figure 8. Peak Error vs Input Frequency with One Cap Averaging

1966fb

A $1\mu F$ capacitor is a good choice for many applications. The peak error at 50Hz/60Hz will be $<1\%$ and the DC error will be $<0.1\%$ with frequencies of 10Hz or more.

Note that both Figure 6 and Figure 8 assume AC-coupled waveforms with a crest factor less than 2, such as sine waves or triangle waves. For higher crest factors and/or AC+DC waveforms, a larger C_{AVE} will generally be required. See Crest Factor and AC + DC Waveforms.

Capacitor Type Selection

The LTC1966 can operate with many types of capacitors. The various types offer a wide array of sizes, tolerances, parasitics, package styles and costs.

Ceramic chip capacitors offer low cost and small size, but are not recommended for critical applications. The value stability over voltage and temperature is poor with many types of ceramic dielectrics. This will not cause an RMS-to-DC accuracy problem except at low frequencies, where it can aggravate the effects discussed in the previous section. If a ceramic capacitor is used, it may be necessary to use a much higher nominal value in order to assure the low frequency accuracy desired.

Another parasitic of ceramic capacitors is leakage, which is again dependent on voltage and particularly temperature. If the leakage is a constant current leak, the $I \cdot R$ drop of the leak multiplied by the output impedance of the LTC1966 will create a constant offset of the output voltage. If the leak is Ohmic, the resistor divider formed with the LTC1966 output impedance will cause a gain error. For $<0.1\%$ gain accuracy degradation, the parallel impedance of the

capacitor leakage will need to be >1000 times the LTC1966 output impedance. Accuracy at this level can be hard to achieve with a ceramic capacitor, particularly with a large value of capacitance and at high temperature.

For critical applications, a film capacitor, such as metallized polyester, will be a much better choice. Although more expensive, and larger for a given value, the value stability and low leakage make metal film capacitors a trouble free choice.

With any type of capacitor, the self resonance of the capacitor can be an issue with the switched capacitor LTC1966. If the self resonant frequency of the averaging capacitor is 1MHz or less, a second smaller capacitor should be added in parallel to reduce the impedance seen by the LTC1966 output stage at high frequencies. A capacitor 100 times smaller than the averaging capacitor will typically be small enough to be a low cost ceramic with a high quality dielectric such as X7R or NPO/COG.

Input Connections

The LTC1966 input is differential and DC coupled. The LTC1966 responds to the RMS value of the differential voltage between Pin 2 and Pin 3, including the DC portion of that difference. However, there is no DC-coupled path from the inputs to ground. Therefore, at least one of the two inputs must be connected with a DC return path to ground.

Both inputs must be connected to something. If either input is left floating, a zero volt output will result.

1966fb

APPLICATIONS INFORMATION

For single-ended DC-coupled applications, simply connect one of the two inputs (they are interchangeable) to the signal, and the other to ground. This will work well for dual supply configurations, but for single supply configurations it will only work well for unipolar input signals. The LTC1966 input voltage range is from rail-to-rail, and when the input is driven above V_{DD} or below V_{SS} (ground for single supply operation) the gain and offset errors will increase substantially after just a few hundred millivolts of overdrive. Fortunately, most single supply circuits measuring a DC-coupled RMS value will include some reference voltage other than ground, and the second LTC1966 input can be connected to that point.

For single-ended AC-coupled applications, Figure 9 shows three alternate topologies. The first one, shown in Figure 9a uses a coupling capacitor to one input while the other is grounded. This will remove the DC voltage difference from the input to the LTC1966, and it will therefore not be part of the resulting output voltage. Again, this connection will work well with dual supply configurations, but in single supply configurations it will be necessary to raise the voltage on the grounded input to assure that the signal at the active input stays within the range of V_{SS} to V_{DD} . If there is already a suitable voltage reference available, connect the second input to that point. If not, a midsupply voltage can be created with two resistors as shown in Figure 9b.

Finally, if the input voltage is known to be between V_{SS} and V_{DD} , it can be AC-coupled by using the configuration shown in Figure 9c. Whereas the DC return path was provided through Pin 3 in Figures 9a and 9b, in this case, the return path is provided on Pin 2, through the input signal voltages. The switched capacitor action between the two input pins of the LTC1966 will cause the voltage

on the coupling capacitor connected to the second input to follow the DC average of the input voltage.

For differential input applications, connect the two inputs to the differential signal. If AC coupling is desired, one of the two inputs can be connected through a series capacitor.

In all of these connections, to choose the input coupling capacitor, C_C , calculate the low frequency coupling time constant desired, and divide by the LTC1966 differential input impedance. Because the LTC1966 input impedance is about 100 times its output impedance, this capacitor is typically much smaller than the output averaging capacitor. Its requirements are also much less stringent, and a ceramic chip capacitor will usually suffice.

Output Connections

The LTC1966 output is differentially, but not symmetrically, generated. That is to say, the RMS value that the LTC1966 computes will be generated on the output (Pin 5) relative to the output return (Pin 6), but these two pins are not interchangeable. For most applications, Pin 6 will be tied to ground (Pin 1), and this will result in the best accuracy. However, Pin 6 can be tied to any voltage between V_{SS} (Pin 4) and V_{DD} (Pin 7) less the maximum output voltage swing desired. This last restriction keeps V_{OUT} itself (Pin 5) within the range of V_{SS} to V_{DD} . If a reference level other than ground is used, it should be a low impedance, both AC and DC, for proper operation of the LTC1966.

Use of a voltage in the range of $V_{DD} - 1V$ to $V_{DD} - 1.3V$ can lead to errors due to the switch dynamics as the NMOS transistor is cut off. For this reason, it is recommended that $OUT\ RTN = 0V$ if V_{DD} is $\leq 3V$.

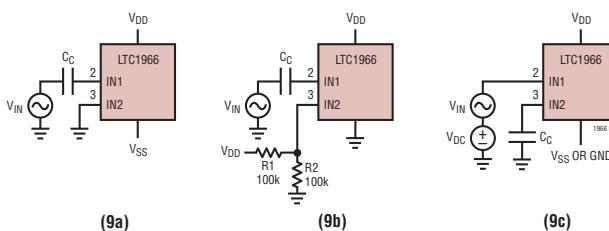


Figure 9. Single-Ended AC-Coupled Input Connection Alternatives

1966fb

LTC1966

APPLICATIONS INFORMATION

In any configuration, the averaging capacitor should be connected between Pins 5 and 6. The LTC1966 RMS DC output will be a positive voltage created at V_{OUT} (Pin 5) with respect to OUT RTN (Pin 6).

Power Supply Bypassing

The LTC1966 is a switched capacitor device, and large transient power supply currents will be drawn as the switching occurs. For reliable operation, standard power supply bypassing must be included. For single supply operation, a 0.01 μ F capacitor from V_{DD} (Pin 7) to GND (Pin 1) located close to the device will suffice. For dual supplies, add a second 0.01 μ F capacitor from V_{SS} (Pin 4) to GND (Pin 1), located close to the device. If there is a good quality ground plane available, the capacitors can go directly to that instead. Power supply bypass capacitors can, of course, be inexpensive ceramic types.

The sampling clock of the LTC1966 operates at approximately 200kHz, and most operations repeat at a rate of 100kHz. If this internal clock becomes synchronized to a multiple or submultiple of the input frequency, significant conversion error could occur. This is particularly important when frequencies exceeding 10kHz can be injected into the LTC1966 via supply or ground bounce. To minimize this possibility, capacitive bypassing is recommended on both supplies with capacitors placed immediately adjacent to the LTC1966. For best results, the bypass capacitors should be separately routed from Pin 7 to Pin 1, and from Pin 4 to Pin 1.

The LTC1966 needs at least 2.7V for its power supply, more for dual supply configurations. The range of allowable negative supply voltages (V_{SS}) vs positive supply voltages (V_{DD}) is shown in Figure 10. Mathematically, the V_{SS} constraint is:

$$-3 \cdot (V_{DD} - 2.7V) \leq V_{SS} \leq GND$$

The LTC1966 has internal ESD absorption devices, which are referenced to the V_{DD} and V_{SS} supplies. For effective in-circuit ESD immunity, the V_{DD} and V_{SS} pins must be connected to a low external impedance. This can be accomplished with low impedance power planes or simply with the recommended 0.01 μ F decoupling to ground on each supply.

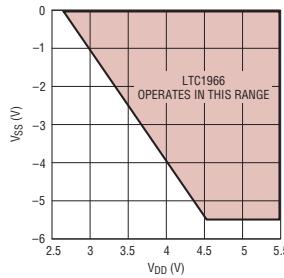


Figure 10. V_{SS} Limits vs V_{DD}

Up and Running!

If you have followed along this far, you should have the LTC1966 up and running by now! Don't forget to enable the device by grounding Pin 8, or driving it with a logic low.

Keep in mind that the LTC1966 output impedance is fairly high, and that even the standard 10M Ω input impedance of a digital multimeter (DMM) or a 10 \times scope probe will load down the output enough to degrade its typical gain error of 0.1%. In the end application circuit, either a buffer or another component with an extremely high input impedance (such as a dual slope integrating ADC) should be used. For laboratory evaluation, it may suffice to use a bench top DMM with the ability to disconnect the 10M Ω shunt.

If you are still having trouble, it may be helpful to skip ahead a few pages and review the Troubleshooting Guide.

What About Response Time?

With a large value averaging capacitor, the LTC1966 can easily perform RMS-to-DC conversion on low frequency signals. It compares quite favorably in this regard to prior generation products because nothing about the $\Delta\Sigma$ circuitry is temperature sensitive. So the RMS result doesn't get distorted by signal driven thermal fluctuations like a log/antilog circuit output does.

However, using large value capacitors results in a slow response time. Figure 11 shows the rising and falling step responses with a 1 μ F averaging capacitor. Although they both appear at first glance to be standard exponential

APPLICATIONS INFORMATION

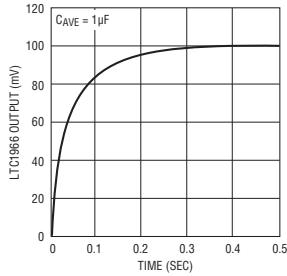
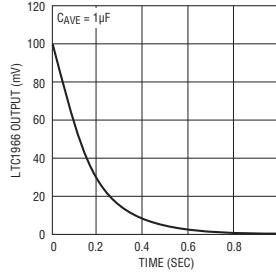
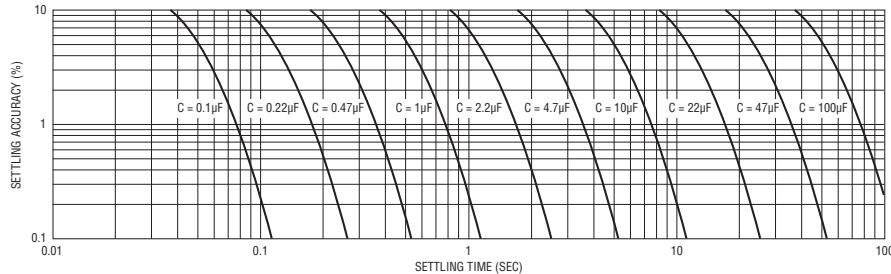
Figure 11a. LTC1966 Rising Edge with $C_{AVE} = 1\mu F$ Figure 11b. LTC1966 Falling Edge with $C_{AVE} = 1\mu F$ 

Figure 12. LTC1966 Settling Time with One Cap Averaging

decay type settling, they are not. This is due to the nonlinear nature of an RMS-to-DC calculation. Also note the change in the time scale between the two; the rising edge is more than twice as fast to settle to a given accuracy. Again this is a necessary consequence of RMS-to-DC calculation.²

Although shown with a step change between 0mV and 100mV, the same response shapes will occur with the LTC1966 for ANY step size. This is in marked contrast to prior generation log/antilog RMS-to-DC converters, whose averaging time constants are dependent on the signal level, resulting in excruciatingly long waits for the output to go to zero.

The shape of the rising and falling edges will be dependent on the total percent change in the step, but for less than the 100% changes shown in Figure 11, the responses will be less distorted and more like a standard exponential

decay. For example, when the input amplitude is changed from 100mV to 110mV (+10%) and back (-10%), the step responses are essentially the same as a standard exponential rise and decay between those two levels. In such cases, the time constant of the decay will be in between that of the rising edge and falling edge cases of Figure 11. Therefore, the worst case is the falling edge response as it goes to zero, and it can be used as a design guide.

Figure 12 shows the settling accuracy vs settling time for a variety of averaging capacitor values. If the capacitor value previously selected (based on error requirements) gives an acceptable settling time, your design is done.

²To convince oneself of this necessity, consider a pulse train of 50% duty cycle between 0mV and 100mV. At very low frequencies, the LTC1966 will essentially track the input. But as the input frequency is increased, the average result will converge to the RMS value of the input. If the rise and fall characteristics were symmetrical, the output would converge to 50mV. In fact though, the RMS value of a 100mV DC-coupled 50% duty cycle pulse train is 70.71mV which as the asymmetrical rise and fall characteristics will converge to as the input frequency is increased.

LTC1966

APPLICATIONS INFORMATION

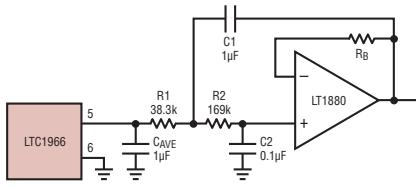
But with $100\mu F$ the settling time to even 10% is a full 38 seconds, which is a long time to wait. What can be done about such a design? If the reason for choosing $100\mu F$ is to keep the DC error with a 75mHz input less than 0.1%, the answer is: not much. The settling time to 1% of 76 seconds is just 5.7 cycles of this extremely low frequency. Averaging very low frequency signals takes a long time.

However, if the reason for choosing $100\mu F$ is to keep the peak error with a 10Hz input less than 0.05%, there is another way to achieve that result with a much improved settling time.

Reducing Ripple with a Post Filter

The output ripple is always much larger than the DC error, so filtering out the ripple can reduce the peak error substantially, without the large settling time penalty of simply increasing the averaging capacitor.

Figure 13 shows a basic 2nd order post filter, for a net 3rd order filtering of the LTC1966 RMS calculation. It uses the $85k\Omega$ output impedance of the LTC1966 as the first resistor of a 3rd order Sallen-Key active RC filter. This topology features a buffered output, which can be desirable depending on the application. However, there are disadvantages to this topology, the first of which is that the op amp input voltage and current errors directly degrade the effective LTC1966 V_{OOS} . The table inset in Figure 13 shows these errors for four of Linear Technology's op amps.



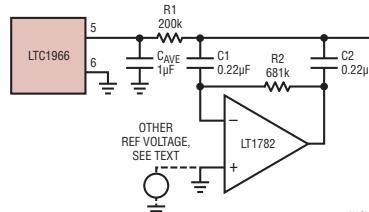
OP AMP	LTC1966	LTC1949	LTC1880	LTC1077	LTC2050
LTC1966 V_{OOS}	$\pm 200\mu V$	$\pm 375\mu V$	$\pm 150\mu V$	$\pm 60\mu V$	$\pm 3\mu V$
$I_{VOS} \cdot R$	$\pm 73\mu V$	$\pm 329\mu V$	$\pm 329\mu V$	$\pm 27\mu V$	
TOTAL OFFSET	$\pm 648\mu V$	$\pm 679\mu V$	$\pm 589\mu V$	$\pm 230\mu V$	
R_g VALUE	294k	SHORT	294k	SHORT	
I_{SQ}	$1\mu A$	$1.2mA$	$48\mu A$	$750\mu A$	

1966 F13

Figure 13. Buffered Post Filter

A second disadvantage is that the op amp output has to operate over the same range as the LTC1966 output, including ground, which in single supply applications is the negative supply. Although the LTC1966 output will function fine just millivolts from the rail, most op amp output stages (and even some input stages) will not. There are at least two ways to address this. First of all, the op amp can be operated split supply if a negative supply is available. Just the op amp would need to do so; the LTC1966 can remain single supply. A second way to address this issue is to create a signal reference voltage a half volt or so above ground. This is most attractive when the circuitry that follows has a differential input, so that the tolerance of the signal reference is not a concern. To do this, tie all three ground symbols shown in Figure 13 to the signal reference, as well as to the differential return for the circuitry that follows.

Figure 14 shows an alternative 2nd order post filter, for a net 3rd order filtering of the LTC1966 RMS calculation. It also uses the $85k\Omega$ output impedance of the LTC1966 as the first resistor of a 3rd order active RC filter, but this topology filters without buffering so that the op amp DC error characteristics do not affect the output. Although the output impedance of the LTC1966 is increased from $85k\Omega$ to $285k\Omega$, this is not an issue with an extremely high input impedance load, such as a dual slope integrating ADC like the ICL7106. And it allows a generic op amp to be used, such as the SOT-23 one shown. Furthermore, it easily works on a single supply rail by tying the noninverting input of the op amp to a low noise reference as optionally shown. This reference will not change the DC voltage at the circuit output, although it does become the AC ground for the filter, thus the (relatively) low noise requirement.



1966 F14

Figure 14. DC Accurate Post Filter

1966fb

APPLICATIONS INFORMATION

Step Responses with a Post Filter

Both of the post filters, shown in Figures 13 and 14, are optimized for additional filtering with clean step responses. The $85\text{k}\Omega$ output impedance of the LTC1966 working into a $1\mu\text{F}$ capacitor forms a 1st order LPF with a -3dB frequency of $\sim 1.8\text{Hz}$. The two filters have $1\mu\text{F}$ at the LTC1966 output for easy comparison with a $1\mu\text{F}$ only case, and both have the same relative (Bessel-like) shape. However, because of the topological differences of pole placements between the various components within the two filters, the net effective bandwidth for Figure 13 is slightly higher ($\approx 1.2 \cdot 1.8 \approx 2.1\text{Hz}$) than with $1\mu\text{F}$ alone, while the bandwidth for Figure 14 is somewhat lower ($\approx 0.7 \cdot 1.8 \approx 1.3\text{Hz}$) than with $1\mu\text{F}$ alone. To adjust the bandwidth of either of them, simply scale all the capacitors by a common multiple, and leave the resistors unchanged.

The step responses of the LTC1966 with $1\mu\text{F}$ only and with the two post filters are shown in Figure 15. This is the rising edge RMS output response to a 10Hz input starting at $t = 0$. Although the falling edge response is the worst case for settling, the rising edge illustrates the ripple that these post filters are designed to address, so the rising edge makes for a better intuitive comparison.

The initial rise of the LTC1966 will have enhanced slew rates with DC and very low frequency inputs due to saturation effects in the $\Delta\Sigma$ modulator. This is seen in Figure 15 in two ways. First, the $1\mu\text{F}$ only output is seen to rise very quickly in the first 40ms . The second way this effect shows up is that the post filter outputs have a modest overshoot, on the order of 3mV to 4mV , or 3% to 4% . This is only

an issue with input frequency bursts at 50Hz or less, and even with the overshoot, the settling to a given level of accuracy improves due to the initial speedup.

As predicted by Figure 6, the DC error with $1\mu\text{F}$ is well under 1mV and is not noticeable at this scale. However, as predicted by Figure 8, the peak error with the ripple from a 10Hz input is much larger, in this case about 5mV . As can be clearly seen, the post filters reduce this ripple. Even the wider bandwidth of Figure 13's filter is seen to cut the ripple down substantially (to $< 1\text{mV}$) while the settling to 1% happens faster. With the narrower bandwidth of Figure 14's filter, the step response is somewhat slower, but the double frequency output ripple is just $180\mu\text{V}$.

Figure 16 shows the step response of the same three cases with a burst of 60Hz rather than 10Hz . With 60Hz , the initial portion of the step response is free of the boost seen in Figure 15 and the two post filter responses have less than 1% overshoot. The $1\mu\text{F}$ only case still has noticeable 120Hz ripple, but both filters have removed all detectable ripple on this scale. This is to be expected; the first order filter will reduce the ripple about 6:1 for a 6:1 change in frequency, while the third order filters will reduce the ripple about 6³:1 or 216:1 for a 6:1 change in frequency.

Again, the two filter topologies have the same relative shape, so the step response and ripple filtering trade-offs of the two are the same, with the same performance of each possible with the other by scaling it accordingly. Figures 17 and 18 show the peak error vs. frequency for a selection of capacitors for the two different filter topologies. To keep the clean step response, scale all three capacitors

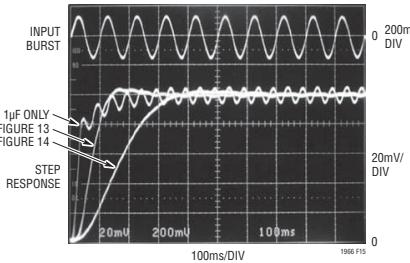


Figure 15. Step Responses with 10Hz Burst

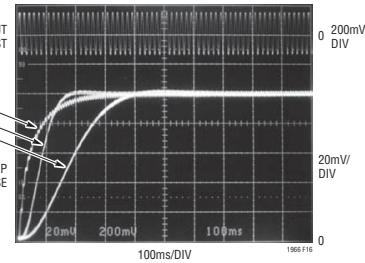


Figure 16. Step Responses with 60Hz Burst

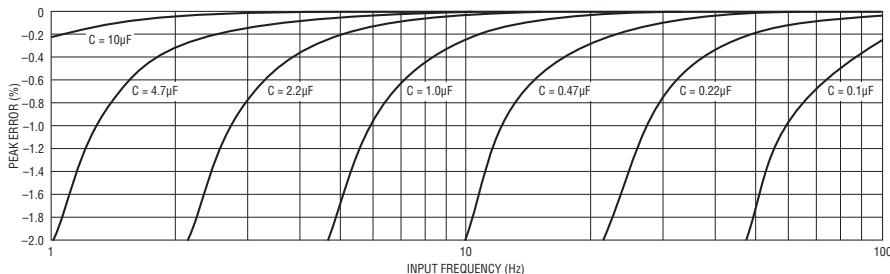
APPLICATIONS INFORMATION

Figure 17. Peak Error vs Input Frequency with Buffered Post Filter

1966 F17

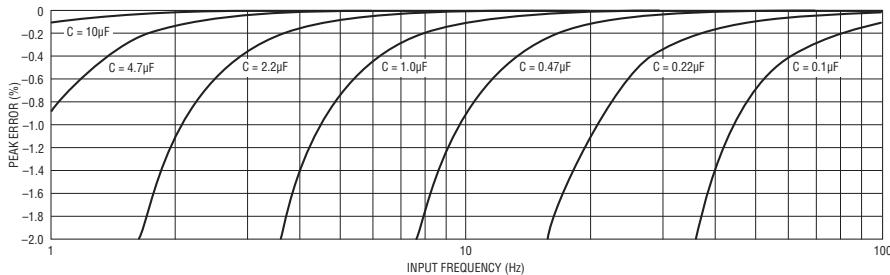


Figure 18. Peak Error vs Input Frequency with DC Accurate Post Filter

1966 F18

within the filter. Scaling the buffered topology of Figure 13 is simple because the capacitors are in a 10:1:10 ratio. Scaling the DC accurate topology of Figure 14 can be done with standard value capacitors; one decade of scaling is shown in Table 2.

Table 2. One Decade of Capacitor Scaling for Figure 14 with EIA Standard Values

C_{AVE}	$C_1 = C_2 =$
1 μF	0.22 μF
1.5 μF	0.33 μF
2.2 μF	0.47 μF
3.3 μF	0.68 μF
4.7 μF	1 μF
6.8 μF	1.5 μF

Figures 19 and 20 show the settling time versus settling accuracy for the buffered and DC accurate post filters, respectively. The different curves represent different scalings of the filters, as indicated by the C_{AVE} value. These are comparable to the curves in Figure 12 (single capacitor case), with somewhat less settling time for the buffered post filter, and somewhat more settling time for the DC accurate post filter. These differences are due to the change in overall bandwidth as mentioned earlier.

The other difference is the settling behavior of the filters below the 1% level. Unlike the case of a 1st order filter, any 3rd order filter can have overshoot and ringing. The filter designs presented here have minimal overshoot and ringing, but are somewhat sensitive to component mismatches. Even the ±12% tolerance of the LTC1966 output impedance can be enough to cause some ringing. The dashed lines indicate what can happen when ±5% capacitors and ±1% resistors are used.

1966fb

APPLICATIONS INFORMATION

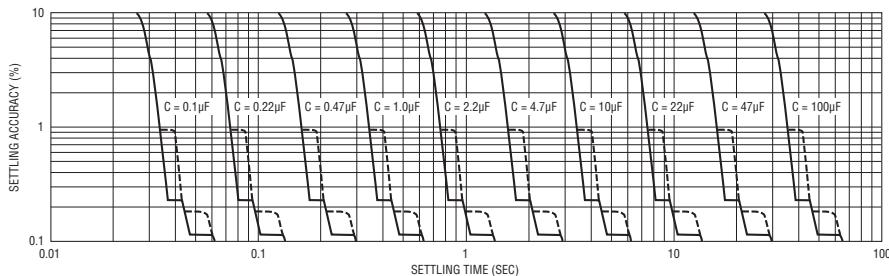


Figure 19. Settling Time with Buffered Post Filter

1066 F14

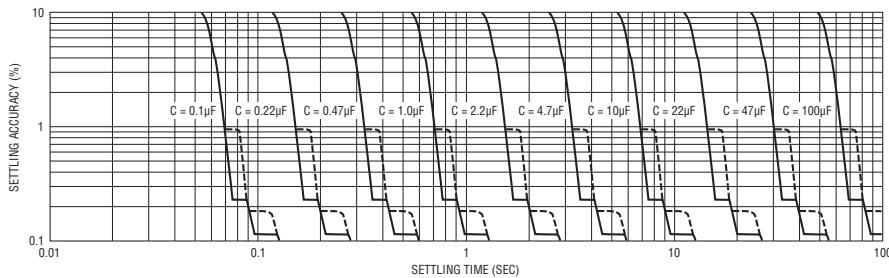


Figure 20. Settling Time with DC Accurate Post Filter

1066 F20

Although the settling times for the post filtered configurations shown on Figures 19 and 20 are not that much different from those with a single capacitor, the point of using a post filter is that the settling times are far better for a given level peak error. The filters dramatically reduce the low frequency averaging ripple with far less impact on settling time.

Crest Factor and AC + DC Waveforms

In the preceding discussion, the waveform was assumed to be AC-coupled, with a modest crest factor. Both assumptions ease the requirements for the averaging capacitor. With an AC-coupled sine wave, the calculation engine squares the input, so the averaging filter that follows is required to filter twice the input frequency, making its job easier. But with a sinewave that includes DC offset, the square of the input has frequency content

at the input frequency and the filter must average out that lower frequency. So with AC + DC waveforms, the required value for C_{AVE} should be based on half of the lowest input frequency, using the same design curves presented in Figures 6, 8, 17 and 18.

Crest factor, which is the peak to RMS ratio of a dynamic signal, also effects the required C_{AVE} value. With a higher crest factor, more of the energy in the signal is concentrated into a smaller portion of the waveform, and the averaging has to ride out the long lull in signal activity. For busy waveforms, such as a sum of sine waves, ECG traces or SCR chopped sine waves, the required value for C_{AVE} should be based on the lowest fundamental input frequency divided as such:

$$f_{DESIGN} = \frac{f_{INPUT(MIN)}}{3 \cdot \sqrt{CF - \sqrt{2}}}$$

1966fb

LTC1966

APPLICATIONS INFORMATION

using the same design curves presented in Figures 6, 8, 17 and 18. For the worst-case of square top pulse trains, that are always either zero volts or the peak voltage, base the selection on the lowest fundamental input frequency divided by twice as much:

$$f_{DESIGN} = \frac{f_{INPUT(MIN)}}{6 \cdot \sqrt{CF - \sqrt{2}}}$$

The effects of crest factor and DC offsets are cumulative. So for example, a 10% duty cycle pulse train from 0V_{PEAK} to 1V_{PEAK} ($CF = \sqrt{10} = 3.16$) repeating at 16.67ms (60Hz) input is effectively only 30Hz due to the DC asymmetry and is effectively only:

$$f_{DESIGN} = \frac{30}{6 \cdot \sqrt{3.16 - \sqrt{2}}} = 3.78\text{Hz}$$

for the purposes of Figures 6, 8, 17 and 18.

Obviously, the effect of crest factor is somewhat simplified above given the factor of 2 difference based on a subjective description of the waveform type. The results will vary somewhat based on actual crest factor and waveform dynamics and the type of filtering used. The above method is conservative for some cases and about right for others.

The LTC1966 works well with signals whose crest factor is 4 or less. At higher crest factors, the internal $\Delta\Sigma$ modulator will saturate, and results will vary depending on the exact frequency, shape and (to a lesser extent) amplitude of the input waveform. The output voltage could be higher or lower than the actual RMS of the input signal.

The $\Delta\Sigma$ modulator may also saturate when signals with crest factors less than 4 are used with insufficient averaging. This will only occur when the output drops to less than 1/4 of the input voltage peak. For instance, a DC-coupled pulse train with a crest factor of 4 has a duty cycle of 6.25% and a 1V_{PEAK} input is 250mV_{RMS}. If this input is 50Hz, repeating every 20ms, and $C_{AVE} = 1\mu\text{F}$, the output will droop during the inactive 93.75% of the waveform. This droop is calculated as:

$$V_{MIN} = \frac{V_{RMS}}{2} \left(1 - e^{-\left(\frac{\text{INACTIVE TIME}}{2 \cdot Z_{OUT} \cdot C_{AVE}} \right)} \right)$$

For the LTC1966, whose output impedance (Z_{OUT}) is 85k Ω , this droop works out to -5.22% , so the output would be reduced to 237mV at the end of the inactive portion of the input. When the input signal again climbs to 1V_{PEAK}, the peak/output ratio is 4.22.

With $C_{AVE} = 10\mu\text{F}$ the droop is only -0.548% to 248.6mV and the peak/output ratio is just 4.022, which the LTC1966 has enough margin to handle without error.

For crest factors less than 3.5, the selection of C_{AVE} as previously described should be sufficient to avoid this droop and modulator saturation effect. But with crest factors above 3.5, the droop should also be checked for each design.

Error Analyses

Once the RMS-to-DC conversion circuit is working, it is time to take a step back and do an analysis of the accuracy of that conversion. The LTC1966 specifications include three basic static error terms, V_{OOS} , V_{IOS} and GAIN. The output offset is an error that simply adds to (or subtracts from) the voltage at the output. The conversion gain of the LTC1966 is nominally 1.000 V_{DCOUT}/V_{RMSIN} and the gain error reflects the extent to which this conversion gain is not perfectly unity. Both of these affect the results in a fairly obvious way.

Input offset on the other hand, despite its conceptual simplicity, effects the output in a nonobvious way. As its name implies, it is a constant error voltage that adds directly with the input. And it is the sum of the input and V_{IOS} that is RMS converted.

This means that the effect of V_{IOS} is warped by the nonlinear RMS conversion. With 0.2mV (typ) V_{IOS} , and a 200mV_{RMS} AC input, the RMS calculation will add the DC and AC terms in an RMS fashion and the effect is negligible:

$$\begin{aligned} V_{OUT} &= \sqrt{(200\text{mV AC})^2 + (0.2\text{mV DC})^2} \\ &= 200.0001\text{mV} \\ &= 200\text{mV} + 1/2\text{ppm} \end{aligned}$$

1966fb

APPLICATIONS INFORMATION

But with 10 \times less AC input, the error caused by V_{IOS} is 100 \times larger:

$$\begin{aligned} V_{OUT} &= \sqrt{(20\text{mV AC})^2 + (0.2\text{mV DC})^2} \\ &= 20.001\text{mV} \\ &= 20\text{mV} + 50\text{ppm} \end{aligned}$$

This phenomena, although small, is one source of the LTC1966's residual nonlinearity.

On the other hand, if the input is DC-coupled, the input offset voltage adds directly. With +200mV and a +0.2mV V_{IOS} , a 200.2mV output will result, an error of 0.1% or 1000ppm. With DC inputs, the error caused by V_{IOS} can be positive or negative depending if the two have the same or opposing polarity.

The total conversion error with a sine wave input using the typical values of the LTC1966 static errors is computed as follows:

$$\begin{aligned} V_{OUT} &= (\sqrt{(500\text{mV AC})^2 + (0.2\text{mV DC})^2}) \cdot 1.001 + 0.1\text{mV} \\ &= 500.600\text{mV} \\ &= 500\text{mV} + 0.120\% \end{aligned}$$

$$\begin{aligned} V_{OUT} &= (\sqrt{(50\text{mV AC})^2 + (0.2\text{mV DC})^2}) \cdot 1.001 + 0.1\text{mV} \\ &= 50.150\text{mV} \\ &= 50\text{mV} + 0.301\% \end{aligned}$$

$$\begin{aligned} V_{OUT} &= (\sqrt{(5\text{mV AC})^2 + (0.2\text{mV DC})^2}) \cdot 1.001 + 0.1\text{mV} \\ &= 5.109\text{mV} \\ &= 5\text{mV} + 2.18\% \end{aligned}$$

As can be seen, the gain term dominates with large inputs, while the offset terms become significant with smaller inputs. In fact, 5mV is the minimum RMS level needed to keep the LTC1966 calculation core functioning normally, so this represents the worst-case of usable input levels.

Using the worst-case values of the LTC1966 static errors, the total conversion error is:

$$\begin{aligned} V_{OUT} &= (\sqrt{(500\text{mV AC})^2 + (0.8\text{mV DC})^2}) \cdot 1.003 + 0.2\text{mV} \\ &= 501.70\text{mV} \\ &= 500\text{mV} + 0.340\% \end{aligned}$$

$$\begin{aligned} V_{OUT} &= (\sqrt{(50\text{mV AC})^2 + (0.8\text{mV DC})^2}) \cdot 1.003 + 0.2\text{mV} \\ &= 50.356\text{mV} \\ &= 50\text{mV} + 0.713\% \end{aligned}$$

$$\begin{aligned} V_{OUT} &= (\sqrt{(5\text{mV AC})^2 + (0.8\text{mV DC})^2}) \cdot 1.003 + 0.2\text{mV} \\ &= 5.279\text{mV} \\ &= 5\text{mV} + 5.57\% \end{aligned}$$

These static error terms are in addition to dynamic error terms that depend on the input signal. See the Design Cookbook for a discussion of the DC conversion error with low frequency AC inputs. The LTC1966 bandwidth limitations cause additional errors with high frequency inputs. Another dynamic error is due to crest factor. The LTC1966 performance versus crest factor is shown in the Typical Performance Characteristics.

Monotonicity and Linearity

The LTC1966, like all implicit RMS-to-DC converters (Figure 3), has a division with the output in the denominator. This works fine most of the time, but when the output is zero or near zero this becomes problematic. The LTC1966 has multiple switched capacitor amplifier stages, and depending on the different offsets and their polarity, the DC transfer curve near zero input can take a few different forms, as shown in the Typical Performance Characteristics graph titled DC Transfer Function Near Zero.

Some units (about 1 of every 16) will even be well behaved with a transfer function that is the upper half of a unit rectangular hyperbola with a focal point on the y-axis of a few millivolts.³ For AC inputs, these units will have a monotonic transfer function all the way down to zero input.

The LTC1966 is trimmed for offsets as small as practical, and the resulting behavior is the best statistical linearity provided the zero region troubles are avoided.

It is possible, and even easy, to force the zero region to be well behaved at the price of additional (though predictable) V_{OOS} and some linearity error. For large enough input signals, this linearity error may be negligible.

³In general, every LTC1966 will have a DC transfer function that is essentially a unit rectangular hyperbola (the gain is not always exactly unity, but the gain error is small) with an X- and Y-offset equal to V_{IOS} and V_{OOS} , respectively, until the inputs are small enough that the delta sigma section gets confused. While some units will be the north half of a north south pair, other units will have two upper halves of the conjugate, east west, hyperbolae. The circuit of Figure 23 will assure a continuous transfer function.

LTC1966

APPLICATIONS INFORMATION

To do this, inject current into the output. As shown in Figure 21, the charge pump output impedance is $170\text{k}\Omega$, with the computational feedback cutting the closed loop output impedance to the $85\text{k}\Omega$ specification. By injecting 30nA of current into this 170Ω , with zero input, a 5mV offset

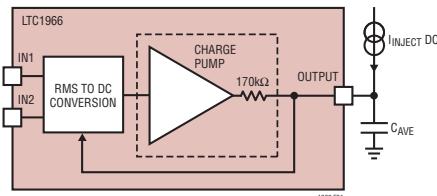


Figure 21. Behavioral Block Diagram of LTC1966

is created at the output feedback point, which is sufficient to overcome the 5mV minimum signal level. With large enough input signals, the computational feedback cuts the output impedance to $85\text{k}\Omega$ so the transfer function asymptotes will have an output offset of 2.5mV , as shown in Figure 22. This is the additional, predictable, V_{OOS} that is added, and should be subtracted from the RMS results, either digitally, or by an analog means.

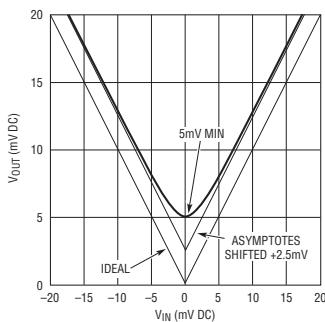


Figure 22a. DC Transfer Function with $I_{INJECT} = 30\text{nA}$

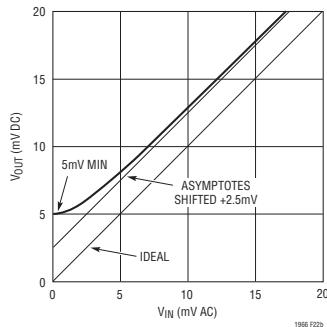


Figure 22b. AC Transfer Function with $I_{INJECT} = 30\text{nA}$

Figure 23 shows an analog implementation of this with the offset and gain errors corrected; only the slight, but necessary, degradation in nonlinearity remains. The circuit works by creating approximately 300mV of bias at the junction of the $10\text{M}\Omega$ resistors when the LTC1966's input/output are zero. The $10\text{M}\Omega$ resistor to the LTC1966 output therefore feeds in 30nA . The loading of this resistor causes a slight reduction in gain which is corrected, as is the nominal 2.5mV offset, by the LT1494 op amp.

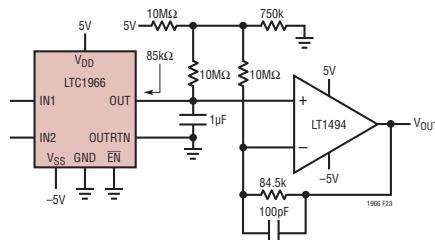


Figure 23. Monotonic AC Response with Offset and Gain Corrected

APPLICATIONS INFORMATION

The two $10M\Omega$ resistors not connected to the supply can be any value as long as they match and the feed voltage is changed for $30nA$ injection. The op amp gain is only 1.00845, so the output is dominated by the LTC1966 RMS results, which keeps errors low. With the values shown, the resistors can be $\pm 2\%$ and only introduce $\pm 170ppm$ of gain error. The 84.5k resistor is the closest match in the 1% EIA values but if the 2% EIA value of 82k were used instead, the gain would only be reduced by 248ppm.

This low error sensitivity is important because the LTC1966 output impedance is $85k\Omega \pm 11.8\%$, which can create a gain error of $\pm 0.1\%$; enough to degrade the overall gain accuracy somewhat. This gain variation term is increased with lower value feed resistors, and decreased with higher value feed resistors.

A bigger error caused by the variation of the LTC1966 output impedance is imperfect cancellation of the output offset introduced by the injected current. The offset correction provided by the LT1494 will be based on a consistent $84.5k\Omega$ times the injected current, while the LTC1966 output impedance will vary enough that the output offset will have a $\pm 300\mu V$ range about the nominal 2.5mV. If this level of output offset is not acceptable, either system calibration or a potentiometer in the LT1494 feedback may be needed.

If the two $10M\Omega$ feed resistors to the LT1494 have significant mismatch, cancellation of the 2.5mV offset would be further impacted, so it is probably worth paying an extra penny or so for 1% resistors or even the better temperature stability of thin film devices. The 300mV feed voltage is not particularly critical because it is nominally cancelled, but the offset errors due to these resistance mismatches is scaled by that voltage.

Note that the input bias current of the op amp used in Figure 23 is also nominally cancelled, but it will add or subtract to the total current injected into the LTC1966 output. With the $1nA$ I_{BIAS} of the LT1494 this is negligible. While it is possible to eliminate the feed resistors by using an op amp with a PNP input stage whose I_{BIAS} is $30nA$

or more, I_{BIAS} is usually only specified for maximum and this circuit needs a minimum of $30nA$, therefore such an approach may not always work.

Because the circuit of Figure 23 subtracts the offset created by the injected current, the LT1494 output with zero LTC1966 input will rest at $+2.5mV$, nominal before offsets, rather than the $5mV$ seen in Figure 22.

Output Errors Versus Frequency

As mentioned in the Design Cookbook, the LTC1966 performs very well with low frequency and very low frequency inputs, provided a large enough averaging capacitor is used.

However, the LTC1966 will have additional dynamic errors as the input frequency is increased. The LTC1966 is designed for high accuracy RMS-to-DC conversion of signals into the audible range. The input sampling amplifiers have a $-3dB$ frequency of 800kHz or so. However, the switched capacitor circuitry samples the inputs at a modest 100kHz nominal. The response versus frequency is depicted in the Typical Performance Characteristics titled Input Signal Bandwidth. Although there is a pattern to the response versus frequency that repeats every sample frequency, the errors are not overwhelming. This is because LTC1966 RMS calculation is inherently wideband, operating properly with minimal oversampling, or even undersampling, using several proprietary techniques to exploit the fact that the RMS value of an aliased signal is the same as the RMS value of the original signal. However, a fundamental feature of the $\Delta\Sigma$ modulator is that sample estimation noise is shaped such that minimal noise occurs with input frequencies much less than the sampling frequency, but such noise peaks when input frequency reaches half the sampling frequency. Fortunately the LTC1966 output averaging filter greatly reduces this error, but the RMS-to-DC topology frequency shifts the noise to low (baseband) frequencies. So with input frequencies above 5kHz to 10kHz, the output will slowly wander around \pm a few percent.

1966fb



LTC1966

APPLICATIONS INFORMATION

Input Impedance

The LTC1966 true RMS-to-DC converter utilizes a 2.5pF capacitor to sample the input at a nominal 100kHz sample frequency. This accounts for the 8MΩ input impedance. See Figure 24 for the equivalent analog input circuit. Note however, that the 8MΩ input impedance does not directly affect the input sampling accuracy. For instance, if a 100k source resistance is used to drive the LTC1966, the sampling action of the input stage will drag down the voltage seen at the input pins with small spikes at every sample clock edge as the sample capacitor is connected to be charged. The time constant of this combination is small, 2.5pF • 100kΩ = 250ns, and during the 2.5μs period devoted to sampling, ten time constants elapse. This allows each sample to settle to within 46ppm and it is these samples that are used to compute the RMS value.

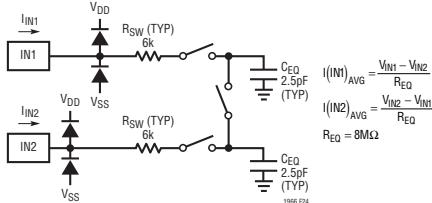


Figure 24. LTC1966 Equivalent Analog Input Circuit

This is a much higher accuracy than the LTC1966 conversion limits, and far better than the accuracy computed via the simplistic resistive divider model:

$$V_{\text{IN}} = V_{\text{SOURCE}} \frac{R_{\text{IN}}}{R_{\text{IN}} + R_{\text{SOURCE}}}$$
$$= V_{\text{SOURCE}} \frac{8\text{M}\Omega}{8\text{M}\Omega + 100\text{k}\Omega}$$
$$= V_{\text{SOURCE}} - 1.25\%$$

This resistive divider calculation does give the correct model of what voltage is seen at the input terminals by a parallel load averaged over a several clock cycles, which is what a large shunt capacitor will do—average the current spikes over several clock cycles.

When high source impedances are used, care must be taken to minimize shunt capacitance at the LTC1966 input so as not to increase the settling time. Shunt capacitance of just 2.5pF will double the input settling time constant and the error in the above example grows from 46ppm to 0.67% (6700ppm). A 13pF scope probe will increase the error to almost 20%. As a consequence, it is important to *not* try to filter the input with large input capacitances unless driven by a low impedance. Keep time constant <<2.5μs.

When the LTC1966 is driven by op amp outputs, whose low DC impedance can be compromised by sharp capacitive load switching, a small series resistor may be added. A 10k resistor will easily settle with the 2.5pF input sampling capacitor to within 1ppm.

These are important points to consider both during design and debug. During lab debug, and even production testing, a high value series resistor to any test point is advisable.

Output Impedance

The LTC1966 output impedance during operation is similarly due to a switched capacitor action. In this case, 59pF of on-chip capacitance operating at 100kHz translates into 170kΩ. The closed loop RMS-to-DC calculation cuts that in half to the nominal 85kΩ specified.

In order to create a DC result, a large averaging capacitor is required. Capacitive loading and time constants are not an issue on the output.

APPLICATIONS INFORMATION

However, resistive loading is an issue and the 10Ω impedance of a DMM or $10\times$ scope probe will drag the output down by -0.85% typ.

During shutdown, the switching action is halted and a fixed 30k resistor shunts V_{OUT} to OUT RTN so that C_{AVE} is discharged.

Guard Ringing the Output

The LTC1966's combination of precision and high output impedance can present challenges that make the use of a guard ring around the output a good idea for many applications.

As mentioned above, a 10M resistive loading to ground will drag down the gain far more than the specified gain tolerance. On a printed circuit board, contaminants from solder flux residue to finger grime can create parasitic resistances, which may be very high impedance, but can have deleterious effects on the realized accuracy. As an example, if the output (Pin 5) is routed near V_{SS} (Pin 4) in a $\pm 5\text{V}$ application, a parasitic resistance of 1G ($1,000\text{M}$) is enough to introduce a $-425\mu\text{V}$ output offset error, more than the specified limit of the LTC1966 itself.

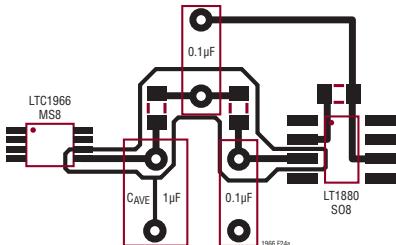


Figure 24a. PCB Layout of Figure 13 with Guard Ring

Use of a guard ring, wherein the LTC1966 output node is completely surrounded by a low impedance voltage, can reduce leakage related errors substantially. The ground ring can be tied to OUTRTN (Pin 6) and should encircle the output (Pin 5), the averaging capacitor terminal, and the destination terminal at the ADC, filter op amp, or whatever else may be next.

Figure 24a shows a sample PCB layout for the circuit of Figure 13, wherein the guard ring trace encloses R1, R2, and the terminals of C1, C2, and the op amp input connected to the high impedance LTC1966 Output. For the circuit of figure 14, the guard ring should enclose R1 and the terminals of C1 and C2, as well as the terminal at the ultimate destination.

Figure 24b shows a sample PCB layout for the circuit of Figure 23. The summing node of the LT1494 has the same high impedance and high accuracy as the LTC1966 output, so here the guard ring encircles both of them. Any leakage between them is benign because the LT1494 forces them to the same nominal voltage.

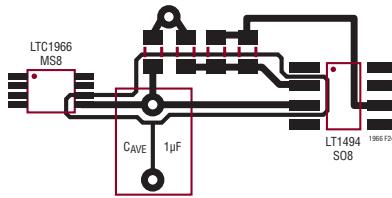


Figure 24b. PCB Layout of Figure 23 with Guard Ring

LTC1966

APPLICATIONS INFORMATION

Interfacing with an ADC

The LTC1966 output impedance and the RMS averaging ripple need to be considered when using an analog-to-digital converter (ADC) to digitize the LTC1966 RMS result.

The simplest configuration is to connect the LTC1966 directly to the input of a type 7106/7136 ADC as shown in Figure 25a. These devices are designed specifically for DVM/DPM use and include display drivers for a 3 1/2 digit LCD segmented display. Using a dual slope conversion, the input is sampled over a long integration window, which results in rejection of line frequency ripple when integration time is an integer number of line cycles. Finally, these parts have an input impedance in the $G\Omega$ range, with specified input leakage of 10pA to 20pA. Such a leakage, combined with the LTC1966 output impedance, results in just 1 μ V to 2 μ V of additional output offset voltage.

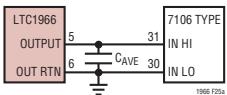


Figure 25a. Interfacing to DVM/DPM ADC

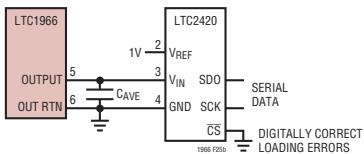


Figure 25b. Interfacing to LTC2420

Another type of ADC that has inherent rejection of RMS averaging ripple is an oversampling $\Delta\Sigma$. With most, but not all, of these devices, it is possible to connect the LTC1966 output directly to the converter input. Issues to look out

for are the input impedance, and any input sampling currents. The input sampling currents drawn by $\Delta\Sigma$ ADCs often have large spikes of current with short durations that can confuse some op amps, but with the large C_{AVG} needed by the LTC1966 these are not an issue.

The average current is important, as it can create LTC1966 errors; if it is constant it will create an offset, while average currents that change with the voltage level create gain errors. Some converters run continuously, others only sample upon demand, and this will change the results in ways that need to be understood. The LTC1966 output impedance has a loose tolerance relative to the usual resistors and the same can be true for the input impedance of $\Delta\Sigma$ ADC, resulting in gain errors from part-to-part. The system calibration techniques described in the following section should be used in applications that demand tight tolerances.

One example of driving an oversampling $\Delta\Sigma$ ADC is shown in Figure 25b. In this circuit, the LTC2420 is used with a 1V V_{REF} . Since the LTC1966 output voltage range is about 1V, and the LTC2420 has a $\pm 12.5\%$ extended input range, this configuration matches the two ranges with room to spare. The LTC2420 has an input impedance of $16.6M\Omega$, resulting in a gain error of -0.4% to -0.6% . In fact, the LTC2420 DC input current is not zero at 0V, but rather at one half its reference, so both an output offset and a gain error will result. These errors will vary from part to part, but with a specific LTC1966 and LTC2420 combination, the errors will be fixed, varying less than $\pm 0.05\%$ over temperature. So a system that has digital calibration can be quite accurate despite the nominal gain and offset error. With 20 bits of resolution, this part is more accurate than the LTC1966, but the extra resolution is helpful because it reduces nonlinearity at the LSB transitions as a digital gain correction is made. Furthermore, its small size and ease of use make it attractive.

APPLICATIONS INFORMATION

As is shown in Figure 25b, where the LTC2420 is set to continuously convert by grounding the CS pin. The gain error will be less if CS is driven at a slower rate, however, the rate should either be consistent or at a rate low enough that the LTC1966 and its output capacitor have fully settled by the beginning of each conversion, so that the loading errors are consistent.

Note that in this circuit, the input current of the LTC2420 is being used to assure monotonicity. The LTC2420 Z_{IN} of $16.6\text{M}\Omega$ is effectively connected to half the reference voltage, so when the LTC1966 has zero signal, $500\text{mV}/16.6\text{M}\Omega = 30\text{nA}$ is provided.

Alternatively, a $5\text{V } V_{REF}$ can be used, but in this case the LTC1966 output span will only use 20% of the LTC2420's input voltage range. Furthermore, if the OUTRTN remains grounded, the injected current with zero signal will be 150nA , resulting in $5\times$ the offset error and nonlinearity shown in Figure 22.

In both of the circuits of Figure 25, a guard ring only has to encircle three terminals, the LTC1966 output, the top of the averaging capacitor, and the ADC input. Figure 26 shows the top copper patterns for example PCB layouts of each.

The low power consumption of the LTC1966 makes it well suited for battery powered applications, and its slow output (DC) makes it an ideal candidate for a micropower ADC.

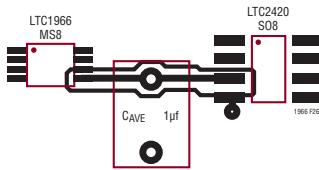


Figure 26b. PCB Layout of Figure 25b with Guard Ring

Figure 10 in Application Note 75, for instance, details a 10-bit ADC with a 35ms conversion time that uses just $29\mu\text{A}$ of supply current. Such an ADC may also be of use within a 4mA to 20mA loop.

Other types of ADCs sample the input signal once and perform a conversion on that one sample. With these ADCs (Nyquist ADCs), a post filter will be needed in most cases to reduce the peak error with low input frequencies. The DC accurate filter of Figure 14 is attractive from an error standpoint, but it increases the impedance at the ADC input. In most cases, the buffered post filter of Figure 13 will be more appropriate for use with Nyquist analog-to-digital converters.

SYSTEM CALIBRATION

The LTC1966 static accuracy can be improved with end system calibration. Traditionally, calibration has been done at the factory, or at a service depot only, typically using manually adjusted potentiometers. Increasingly, systems are being designed for electronic calibration where the accuracy corrections are implemented in digital code wherever possible, and with calibration DACs where necessary. Additionally, many systems are now designed for self calibration, in which the calibration occurs inside the machine, automatically without user intervention.

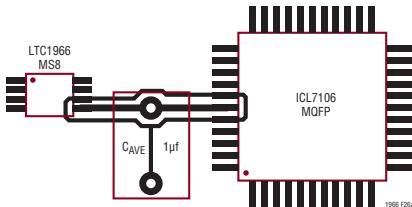


Figure 26a. PCB Layout of Figure 25a with Guard Ring

LTC1966

APPLICATIONS INFORMATION

Whatever calibration scheme is used, the linearity of the LTC1966 will improve the calibrated accuracy over that achievable with older log/antilog RMS-to-DC converters. Additionally, calibration using DC reference voltages are essentially as accurate with the LTC1966 as those using AC reference voltages. Older log/antilog RMS-to-DC converters required nonlinear input stages (rectifiers) whose linearity would typically render DC based calibration unworkable.

The following are four suggested calibration methods. Implementations of the suggested adjustments are dependent on the system design, but in many cases, gain and output offset can be corrected in the digital domain, and will include the effect of all gains and offsets from the LTC1966 output through the ADC. Input offset voltage, on the other hand, will have to be corrected with adjustment to the actual analog input to the LTC1966.

The methods below assume the unaltered linearity of the LTC1966, i.e. without the monotonicity fix of Figure 21. If this is present, the V_{OOS} shift it introduces should be taken out before using either method for which V_{OOS} is not calibrated. Also, the nonlinearity it introduces will increase the 20mV readings discussed below by 0.78% but increase the 200mV readings only 78ppm. There are a variety of ways to deal with these errors, including possibly ignoring them, but the specifics will depend on system requirements. Designers are cautioned to avoid the temptation to digitally take out the hyperbolic transfer function introduced because if the offsets are not exactly the nominals assumed, the system will end up right back where it began with a potential discontinuity with zero input, either from a divide by zero or from a square root of a negative number in the calculations to undo the hyperbolic transfer function. An adaptive algorithm would most likely be necessary to safely take out more than half of the introduced nonlinearity.

If a 5V reference is used in the connection of Figure 25b, the V_{OOS} and nonlinearity created would be even larger,

and will no doubt be more tempting to correct for. Designers are likewise cautioned against correcting for all of the nonlinearity.

AC-Only, 1 Point

The dominant error at full-scale will be caused by the gain error, and by applying a full-scale sine wave input, this error can be measured and corrected for. Unlike older log/antilog RMS-to-DC converters, the correction should be made for zero error at full scale to minimize errors throughout the dynamic range.

The best frequency for the calibration signal is roughly ten times the -0.1% DC error frequency. For $1\mu F$, -0.1% DC error occurs at 8Hz, so 80Hz is a good calibration frequency, although anywhere from 60Hz to 100Hz should suffice.

The trade-off here is that on the one hand, the DC error is input frequency dependent, so a calibration signal frequency high enough to make the DC error negligible should be used. On the other hand, as low a frequency as can be used is best to avoid attenuation of the calibrated AC signal, either from parasitic RC loading or insufficient op amp gain. For instance, with a 1kHz calibration signal, a 1MHz op amp will typically only have 60dB of open loop gain, so it could attenuate the calibration signal a full 0.1%.

AC-Only, 2 Point

The next most significant error for AC-coupled applications will be the effect of output offset voltage, noticeable at the bottom end of the input scale. This too can be calibrated out if two measurements are made, one with a full-scale sine wave input and a second with a sine wave input (of the same frequency) at 10% of full-scale. The trade-off in selecting this second level is that it should be small enough that the gain error effect becomes small compared to the gain error effect at full-scale, while on the other hand, not using so small an input that the input offset voltage becomes an issue.

APPLICATIONS INFORMATION

The calculations of the error terms for a 200mV full-scale case are:

$$\text{Gain} = \frac{\text{Reading at } 200\text{mV} - \text{Reading at } 20\text{mV}}{180\text{mV}}$$

$$\text{Output Offset} = \frac{\text{Reading at } 20\text{mV}}{\text{Gain}} - 20\text{mV}$$

DC, 2 Point

DC based calibration is preferable in many cases because a DC voltage of known, good accuracy is easier to generate than such an AC calibration voltage. The only down side is that the LTC1966 input offset voltage plays a role. It is therefore suggested that a DC based calibration scheme check at least two points: \pm full-scale. Applying the $-$ full-scale input can be done by physically inverting the voltage or by applying the same $+$ full-scale input to the opposite LTC1966 input.

For an otherwise AC-coupled application, only the gain term may be worth correcting for, but for DC-coupled applications, the input offset voltage can also be calculated and corrected for.

The calculations of the error terms for a 200mV full-scale case are:

$$\text{Gain} = \frac{\text{Reading at } 200\text{mV} + \text{Reading at } -200\text{mV}}{400\text{mV}}$$

$$\text{Input Offset} = \frac{\text{Reading at } -200\text{mV} - \text{Reading at } 200\text{mV}}{2 \cdot \text{Gain}}$$

Note: Calculation of and correction for input offset voltage are the only way in which the two LTC1966 inputs (IN1, IN2) are distinguishable from each other. The calculation above assumes the standard definition of offset; that a positive offset is the case of a positive voltage error inside the device that must be corrected by applying a like negative voltage outside. The offset is referred to whichever pin is driven positive for the $+$ full-scale reading.

DC, 3 Point

One more point is needed with a DC calibration scheme to determine output offset voltage: $+10\%$ of full scale.

The calculation of the input offset is the same as for the 2-point calibration above, while the gain and output offset are calculated for a 200mV full-scale case as:

$$\text{Gain} = \frac{\text{Reading at } 200\text{mV} - \text{Reading at } 20\text{mV}}{180\text{mV}}$$

$$\text{Output Offset} = \frac{\text{Reading at } 200\text{mV} + \text{Reading at } -200\text{mV} - 400\text{mV} \cdot \text{Gain}}{2}$$

LTC1966

APPLICATIONS INFORMATION

TROUBLESHOOTING GUIDE

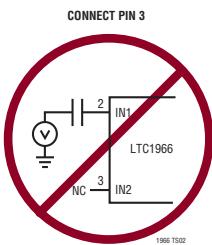
Top Ten LTC1966 Application Mistakes

1. Circuit won't work—Dead On Arrival—no power drawn.
– Probably forgot to enable the LTC1966 by pulling Pin 8 low.

Solution: Tie Pin 8 to Pin 1.

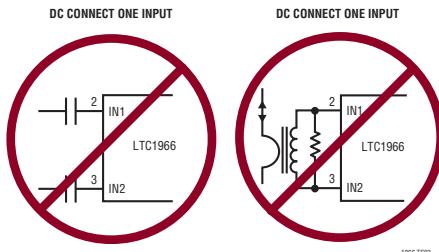
2. Circuit won't work, but draws power. Zero or very little output, single-ended input application.
– Probably didn't connect both input pins.

Solution: Tie both inputs to something. See Input Connections in the Design Cookbook.



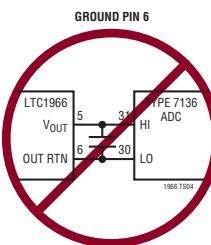
3. Screwy results, particularly with respect to linearity or high crest factors; differential input application.
– Probably AC-coupled both input pins.

Solution: Make at least one input DC-coupled. See Input Connections in the Design Cookbook.



4. Gain is low by a few percent, along with other screwy results.
– Probably tried to use output in a floating, differential manner.

Solution: Tie Pin 6 to a low impedance. See Output Connections in the Design Cookbook.



5. Offsets perceived to be out of specification because $0V \neq 0V$ out.
– The offsets are not specified at $0V$ in. No RMS-to-DC converter works well at 0 due to a divide-by-zero calculation.

Solution: Measure V_{IOS}/V_{OOS} by extrapolating readings > $\pm 5mV_{DC}$.

6. Linearity perceived to be out of specification particularly with small input signals.
– This could again be due to using $0V$ in as one of the measurement points.

Solution: Check Linearity from $5mV_{RMS}$ to $500mV_{RMS}$.

- The input offset voltage can cause small AC linearity errors at low input amplitudes as well. See Error Analyses section.

Possible Solution: Include a trim for input offset.

APPLICATIONS INFORMATION

7. Output is noisy with >10kHz inputs.
 - This is a fundamental characteristic of this topology. The LTC1966 is designed to work very well with inputs of 1kHz or less. It works okay as high as 1MHz, but it is limited by aliased $\Delta\Sigma$ noise.

Solution: Bandwidth limit the input or digitally filter the resulting output.
8. Large errors occur at crest factors approaching, but less than 4.
 - Insufficient averaging.

Solution: Increase C_{AVE} . See Crest Factor and AC + DC Waveforms section for discussion of output droop.
9. Screwy results, errors > spec limits, typically 1% to 5%.
 - High impedance ($85k\Omega$) and high accuracy (0.1%) require clean boards! Flux residue, finger grime, etc. all wreak havoc at this level.

Solution: Wash the board.

Helpful Hint: Sensitivity to leakages can be reduced significantly through the use of guard traces.

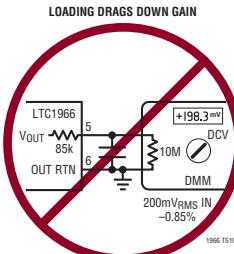


10. Gain is low by $\leq 1\%$ or more, no other problems.
 - Probably due to circuit loading. With a DMM or a 10x scope probe, $Z_{IN} = 10M\Omega$. The LTC1966 output is $85k\Omega$, resulting in -0.85% gain error. Output impedance is higher with the DC accurate post filter.

Solution: Remove the shunt loading or buffer the output.

 - Loading can also be caused by cheap averaging capacitors.

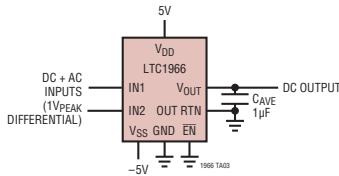
Solution: Use a high quality metal film capacitor for C_{AVE} .



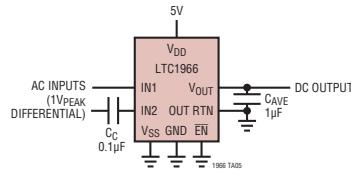
LTC1966

TYPICAL APPLICATIONS

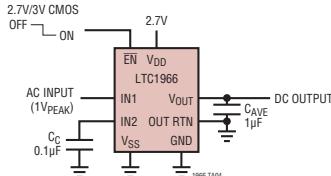
±5V Supplies, Differential, DC-Coupled RMS-to-DC Converter



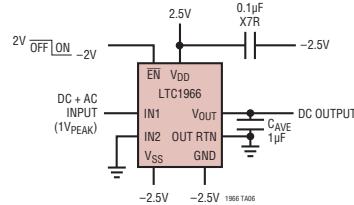
5V Single Supply, Differential, AC-Coupled RMS-to-DC Converter



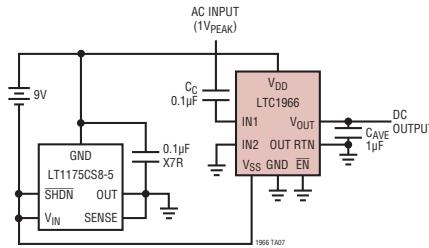
2.7V Single Supply, Single Ended, AC-Coupled RMS-to-DC Converter with Shutdown



±2.5V Supplies, Single Ended, DC-Coupled RMS-to-DC Converter with Shutdown

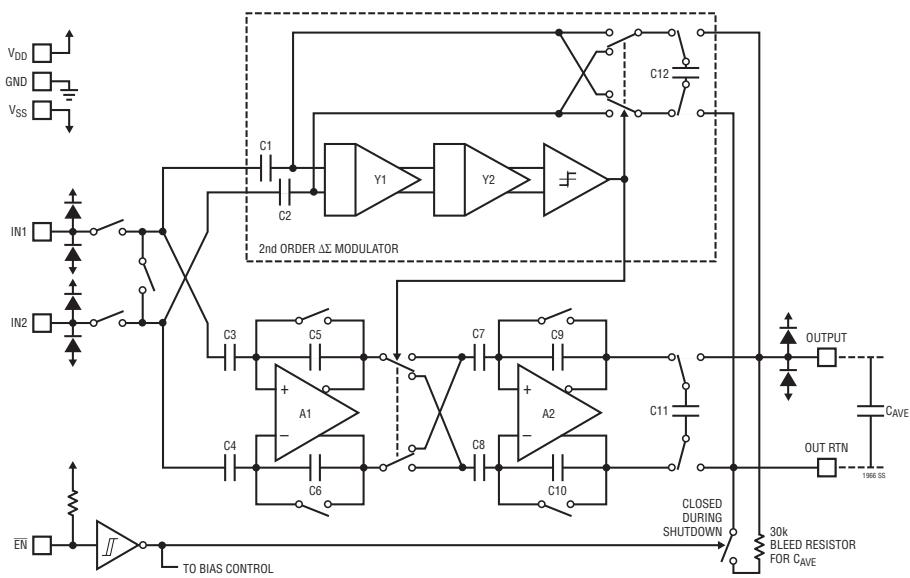


Battery Powered Single-Ended AC-Coupled RMS-to-DC Converter



1966fb

SIMPLIFIED SCHEMATIC

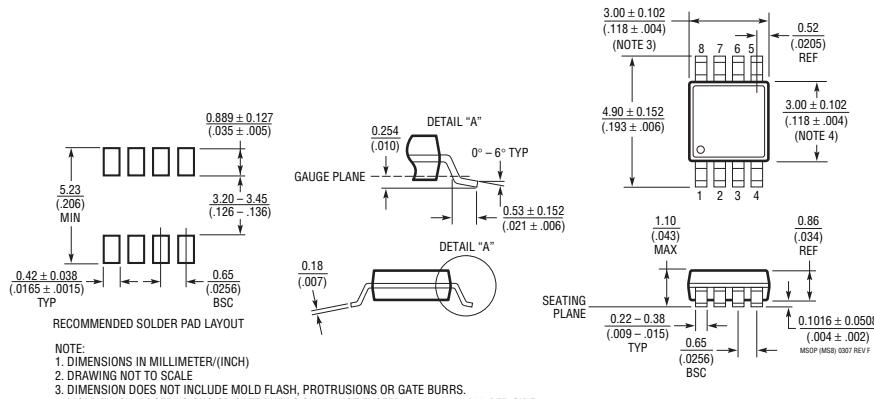


PACKAGE DESCRIPTION

MS8 Package

8-Lead Plastic MSOP

(Reference LTC DWG # 05-08-1660 Rev F)



NOTE:

1966fb

REVISION HISTORY (Revision history begins at Rev B)

REV	DATE	DESCRIPTION	PAGE NUMBER
B	5/11	Revised entire data sheet to add H- and MP- grades	1 to 38



Information furnished by Linear Technology Corporation is believed to be accurate and reliable. However, no responsibility is assumed for its use. Linear Technology Corporation makes no representation that the interconnection of its circuits as described herein will not infringe on existing patent rights.

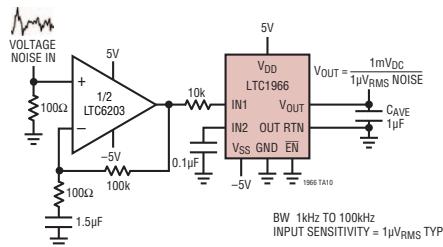
1966fb

37

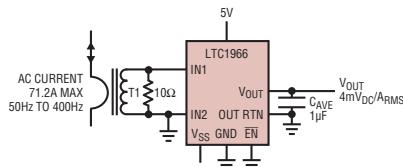
LTC1966

TYPICAL APPLICATION

RMS Noise Measurement

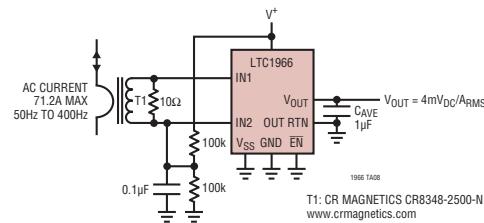


70A Current Measurement



1966 TA09

Single Supply RMS Current Measurement



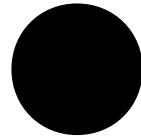
1966 TA08

RELATED PARTS

PART NUMBER	DESCRIPTION	COMMENTS
LT [®] 1077	Micropower, Single Supply Precision Op Amp	48µA I _{SY} , 60µV V _{OS(MAX)} , 450pA I _{OS(MAX)}
LT1175-5	Negative, -5V Fixed, Micropower LDO Regulator	45µA I _Q . Available in SO-8 or SOT-23
LT1494	1.5µA Max, Precision Rail-to-Rail I/O Op Amp	375µV V _{OS(MAX)} , 100pA I _{OS(MAX)}
LT1782	General Purpose SOT-23 Rail-to-Rail Op Amp	40µA I _{SY} , 800µV V _{OS(MAX)} , 2nA I _{OS(MAX)}
LT1880	SOT-23 Rail-to-Rail Output Precision Op Amp	1.2mA I _{SY} , 150µV V _{OS(MAX)} , 900pA I _{OS(MAX)}
LTC1967	Precision, Extended Bandwidth RMS to DC Converter	330µA I _{SY} , ΔΣ RMS Conversion to 4MHz
LTC1968	Precision, Wide Bandwidth RMS to DC Converter	2.3mA I _{SY} , ΔΣ RMS Conversion to 15MHz
LTC2050	Zero Drift Op Amp in SOT-23	750µA I _{SY} , 3µV V _{OS(MAX)} , 75pA I _{B(MAX)}
LT2178/LT2178A	17µA Max, Single Supply Precision Dual Op Amp	14µA I _{SY} , 120µV V _{OS(MAX)} , 350pA I _{OS(MAX)}
LTC2402	2-Channel, 24-bit, Micropower, No Latency ΔΣ™ ADC	200µA I _{SY} , 4ppm INL, 10ppm TUE
LTC2420	20-bit, Micropower, No Latency ΔΣ ADC in SO-8	200µA I _{SY} , 8ppm INL, 16ppm TUE
LTC2422	2-Channel, 20-bit, Micropower, No Latency ΔΣ ADC	Dual Channel Version of LTC2420

1966fb

Document Information



Project reference

317947, FP7-ICT-2011-8

Context

Deliverable 2.1, workpackage 2

Document title

201309_Arduino_Sands_Deliverable_2-1_Mosfet

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

1. Sep. 2013

Table of contents

Document Information.....	1
Table of contents.....	2
Summary.....	3
Purpose within SandS.....	3
SandS Mosfet module.....	4
AVR core.....	4
Power the board.....	4
Mosfet.....	4
Hardware.....	5
Board dimensions.....	6
Mounting considerations.....	6
Schematic.....	6
Board file.....	6
Pin layout.....	6
Firmware.....	6
Licenses.....	7
Hardware.....	7
Firmware.....	7
Appendix 1 - Schematics.....	8
Appendix 2 - Datasheets.....	10

Summary

This document describes the main characteristics of the SandS Mosfet module, an open source hardware - Arduino compatible board, exposing a 5V serial port, and a SandS-enhanced I2C port. This board runs a series of commands to control its on-board mosfet.

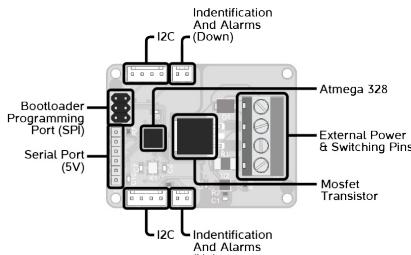


Image 1: block description, Mosfet module

The use scenario for this board is to be switching on and off the power or pulsing a signal on a DC motor, heating element, or similar. The voltage allowance of the mosfet is 75 VDC, while the current is 130 A.

The board will as well estimate the current consumption of the device my measuring it with an on-board operational amplifier.

The board runs from the power provided by the I2C port. It can be reprogrammed from the Arduino IDE using a USB-Serial cable and the SandS special library. It comes with a pre-loaded firmware and runs a modified version of the I2C protocol that allows automatic identification. In other words, it is possible to dynamically assign addresses to the board.

Even if the board has been specifically designed for the SandS project, the pins exposed directly from the processor (the ATmega328) can be addressed as Arduino digital pins.

Purpose within SandS

The purpose of this module is to run DC motors in devices like mixers, or turn on/off components like lamps, heaters or similar.

SandS Mosfet module

The SandS mosfet module is a microcontroller board operating as a I2C peripheral to the SandS motherboard. It is made of two blocks:

- a microcontroller (the ATmega328): running at 16MHz it listens to the I2C port and executes the different operations on the module
- a mosfet transistor used to switch/pulse the power on any external devices running DC voltages up to 75 V and current up to 130 A.

It comes with two connections for the SandS enhanced I2C (two-wire interface + IDENT + ALARM), one 5V RS-232 port, and the SPI programming port to burn the processor's bootloader. See Image 1 for a block diagram of the board.

AVR core

The AVR core chosen for the SandS mosfet module is the ATmega328, which main characteristics are described in Table 1:

Processor	ATmega328
Flash Memory	32 KB of which 512 B used by bootloader
SRAM	2 KB
EEPROM	1 KB
Operating Voltage	5 V
Hardware Serial ports	1
Hardware SPI ports	1
Hardware I2C ports	1

Table 1: main characteristics of the AVR core

This processor can be programmed as an Arduino UNO board directly from the standard Arduino IDE, there is more information how to program the board on the *Getting Started* guide to the module.

The ATmega328 does not have an internal USB peripheral, therefore, the SandS mosfet module needs using a USB-Serial cable, as a way to achieve USB communication.

Power the board

Input Voltage (recommended)	5 V through the I2C connector
Current (typ)	50 mA

Table 2: powering the board

All the SandS modules are powered via the I2C cable. However, during reprogramming operations, it is possible to power the modules through the 5V serial connector.

Mosfet

Voltage (max)	75 VDC
Current (max)	130 A
RDS (typ)	5 mOhm
Power (max)	250 W

Table 3: main characteristics of the mosfet

The mosfet chosen is the IRFS3307, for further information, it is recommended to check its datasheet attached at Appendix 2.

Hardware

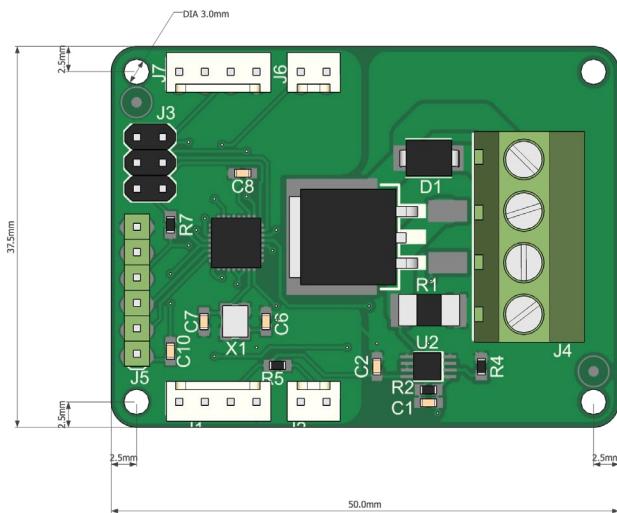


Image 2: Top view, Sands mosfet module v0002

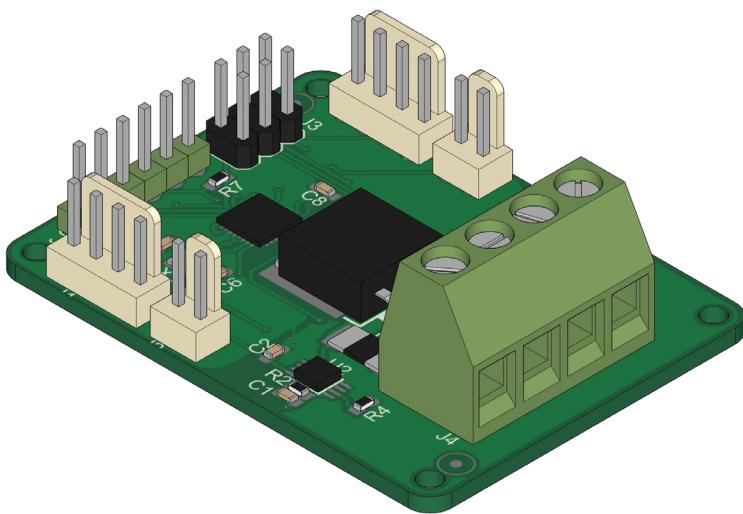


Image 3: Isometric view of the SandS mosfet module v0002

Board dimensions

The board has been designed to easily accommodate the parts in a 2-layer design in the size of 37,5x50mm. It comes with 3mm holes at each corner, making it easy to mount it using standard plastic spacers and M3 screws.

Check Image 2 for further details about measurements and location of the holes on the board.

Image 3 gives you an idea of the proportions of the different components on the board.

Mounting considerations

It is recommended to mount the board in a plastic box. The mosfet module is meant to be switching/pulsing mid DC voltage, a metallic box could provoke a shortcircuit that would damage the board and be potentially harmful.

The module will be connected as part of a I2C bus. This means that the board will use standard board-to-cable connectors both for the uplink and the downlink of the I2C and therefore it is recommended leaving space for those cables to exit the box. Alternatively it is possible to mount differen types of connectors on the plastic housing and simply extend the board with those.

For safety purposes, the mosfet connects through a screw connector. If you were in the need to use a different type of connector, you should fix it to the board's box and plug it in to the screw connector.

Schematic

The board's schematic shows all the components on it. You can check all the schematics at Appendix 1, at the end of this document.

The schematics come as a single document showing:

- a mosfet being switched from a pin on the microcontroller
- an operational amplifier to make current measurements
- Microcontroller: is the ATmega328, with a 16Mhz crystal
- the SPI connector for reprogramming the processor's bootloader
- the I2C uplink and downlink connectors
- the IDENT and ALARM connectors
- the serial port connection for reprogramming the device and debugging it's firmware

Board file

The board file is a 2-layered design with the components

separated strategically to avoid potential interferences.

Half of the board is reserved for the mosfet, while the other half is dedicated to the AVR core and all the connectors.

Images 2 and 3 give a pretty good idea of this layout. Both schematic and board file for this design come bundled in the same zipped file as this document and can be modified using the Eagle Cad software.

Pin layout

Image 4 shows the pin layout for the board as well as the description of each one of the pins separately.

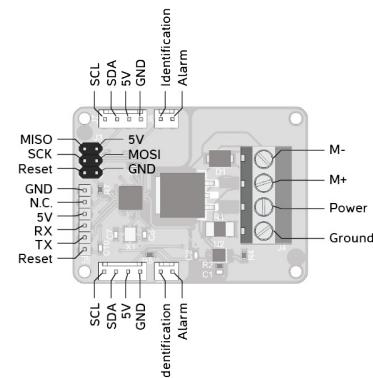


Image 4: pin layout, Mosfet module

Firmware

The SandS mosfet board comes with a preprogrammed firmware that will get the processor to receive commands through the I2C protocol and execute them accordingly. The normal operation mode doesn't need of reprogramming the device. All the specific commands for this device are described in depth in the TWI Protocol document that comes bundled with this one.

The firmware is open source and comes installed in the board. It is possible to reprogram it directly from the Arduino IDE. It is also possible to use other tools like command line, or AVR studio to change the firmware in the processor.

Licenses

Hardware

The design of the boards falls under the CERN Open Hardware License 1.2, for more information refer to the following link:

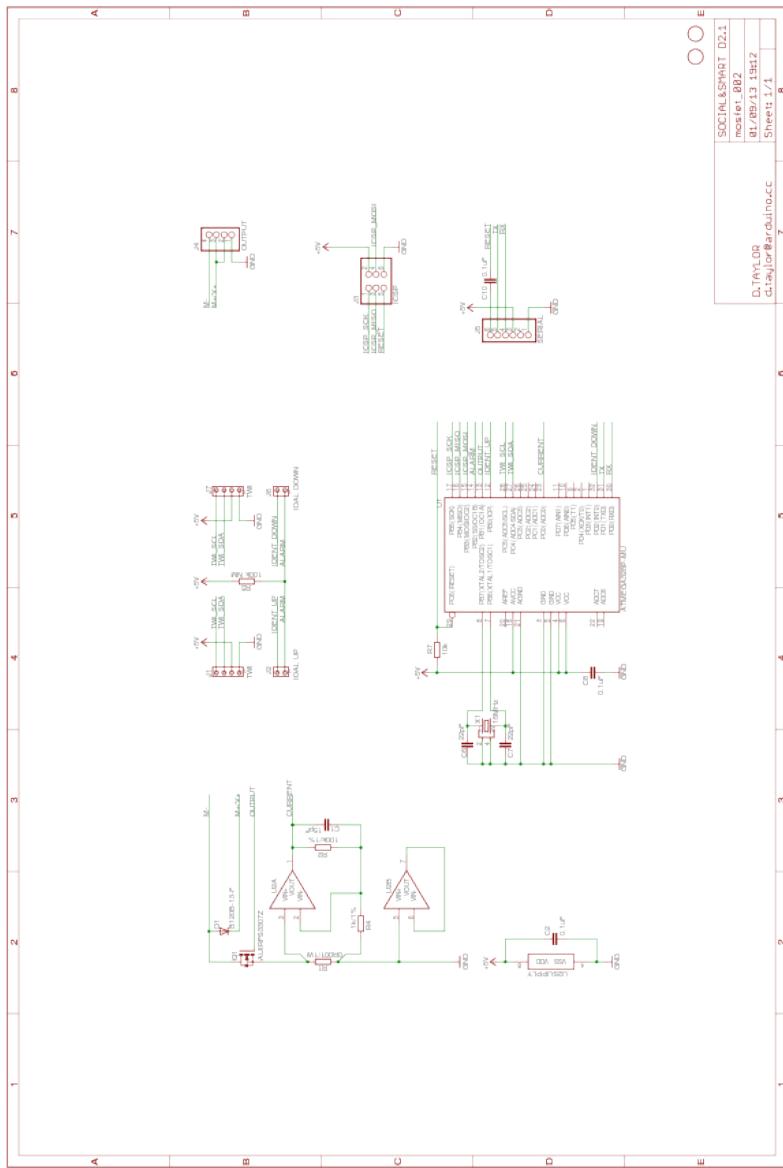
http://ohwr.org/attachments/2388/cern_ohl_v_1_2.txt

Firmware

The firmware for the ATmega328 processor used in the SandS Mosfet board is based on the Arduino core and therefore it is licensed under LGPL, however all the source is available for anybody to use. To read more about this license, refer to:

<https://www.gnu.org/licenses/lgpl.html>

Appendix 1 - Schematics



Appendix 2 - Datasheets

IRFB3307

IRFS3307

IRFSL3307

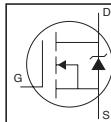
Applications

- High Efficiency Synchronous Rectification in SMPS
- Uninterruptible Power Supply
- High Speed Power Switching
- Hard Switched and High Frequency Circuits

Benefits

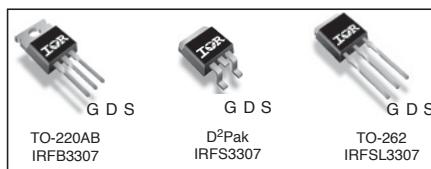
- Improved Gate, Avalanche and Dynamic dV/dt Ruggedness
- Fully Characterized Capacitance and Avalanche SOA
- Enhanced body diode dV/dt and dI/dt Capability

HEXFET® Power MOSFET



V_{DSS}	75V
R_{DS(on)}	typ. 5.0mΩ
	max. 6.3mΩ

I_D **130A**



Absolute Maximum Ratings

Symbol	Parameter	Max.	Units
I _D @ T _C = 25°C	Continuous Drain Current, V _{GS} @ 10V	130①	A
I _D @ T _C = 100°C	Continuous Drain Current, V _{GS} @ 10V	91①	
I _{DM}	Pulsed Drain Current ②	510	
P _D @ T _C = 25°C	Maximum Power Dissipation	250	W
	Linear Derating Factor	1.6	W/°C
V _{GS}	Gate-to-Source Voltage	± 20	V
dV/dt	Peak Diode Recovery ④	11	V/ns
T _J	Operating Junction and Storage Temperature Range	-55 to + 175	°C
T _{STG}	Soldering Temperature, for 10 seconds (1.6mm from case)	300	
	Mounting torque, 6-32 or M3 screw	10lb·in (1.1N·m)	

Avalanche Characteristics

E _{AS} (Thermally limited)	Single Pulse Avalanche Energy ③	270	mJ
I _{AR}	Avalanche Current ①	See Fig. 14, 15, 16a, 16b	A
E _{AR}	Repetitive Avalanche Energy ⑤		mJ

Thermal Resistance

Symbol	Parameter	Typ.	Max.	Units
R _{lJC}	Junction-to-Case ⑨	—	0.61	
R _{lCS}	Case-to-Sink, Flat Greased Surface , TO-220	0.50	—	
R _{lJA}	Junction-to-Ambient, TO-220 ⑨	—	62	°C/W
R _{lJA}	Junction-to-Ambient (PCB Mount) , D ² Pak ⑨⑩	—	40	

Static @ $T_J = 25^\circ\text{C}$ (unless otherwise specified)

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
$V_{(\text{BR})DSS}$	Drain-to-Source Breakdown Voltage	75	—	—	V	$V_{GS} = 0V, I_D = 250\mu\text{A}$
$\Delta V_{(\text{BR})DSS}/\Delta T_J$	Breakdown Voltage Temp. Coefficient	—	0.069	—	V/ $^\circ\text{C}$	Reference to 25°C , $I_D = 1\text{mA}$ ②
$R_{DS(on)}$	Static Drain-to-Source On-Resistance	—	5.0	6.3	$\text{m}\Omega$	$V_{GS} = 10V, I_D = 75\text{A}$ ③
$V_{GS(\text{th})}$	Gate Threshold Voltage	2.0	—	4.0	V	$V_{DS} = V_{GS}, I_D = 150\mu\text{A}$
I_{DSS}	Drain-to-Source Leakage Current	—	—	20	μA	$V_{DS} = V_{GS}, I_D = 0\text{V}$
I_{GSS}	Gate-to-Source Forward Leakage	—	—	200	nA	$V_{GS} = 20V$
	Gate-to-Source Reverse Leakage	—	—	-200	nA	$V_{GS} = -20V$
R_G	Gate Input Resistance	—	1.5	—	Ω	$f = 1\text{MHz}$, open drain

Dynamic @ $T_J = 25^\circ\text{C}$ (unless otherwise specified)

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
g_{fs}	Forward Transconductance	98	—	—	S	$V_{DS} = 50V, I_D = 75\text{A}$
Q_g	Total Gate Charge	—	120	180	nC	$I_D = 75\text{A}$
Q_{gs}	Gate-to-Source Charge	—	35	—	nC	$V_{DS} = 60V$
Q_{gd}	Gate-to-Drain ("Miller") Charge	—	46	—	nC	$V_{GS} = 10V$ ⑤
$t_{d(on)}$	Turn-On Delay Time	—	26	—	ns	$V_{DD} = 48V$
t_r	Rise Time	—	120	—	ns	$I_D = 75\text{A}$
$t_{d(off)}$	Turn-Off Delay Time	—	51	—	ns	$R_G = 3.9\Omega$
t_f	Fall Time	—	63	—	ns	$V_{GS} = 10V$ ⑤
C_{iss}	Input Capacitance	—	5150	—	pF	$V_{GS} = 0V$
C_{oss}	Output Capacitance	—	460	—	pF	$V_{DS} = 50V$
C_{oss}	Reverse Transfer Capacitance	—	250	—	pF	$f = 1.0\text{MHz}$
C_{oss} eff. (ER)	Effective Output Capacitance (Energy Related)	—	570	—	pF	$V_{GS} = 0V, V_{DS} = 0V$ to $60V$ ⑦, See Fig. 11
C_{oss} eff. (TR)	Effective Output Capacitance (Time Related)⑥	—	700	—	pF	$V_{GS} = 0V, V_{DS} = 0V$ to $60V$ ⑥, See Fig. 5

Diode Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Conditions
I_s	Continuous Source Current (Body Diode)	—	—	130①	A	MOSFET symbol showing the integral reverse p-n junction diode.
I_{SM}	Pulsed Source Current (Body Diode) ②	—	—	510	A	
V_{SD}	Diode Forward Voltage	—	—	1.3	V	$T_J = 25^\circ\text{C}, I_S = 75\text{A}, V_{GS} = 0V$ ⑤
t_{rr}	Reverse Recovery Time	—	38	57	ns	$T_J = 25^\circ\text{C}$ $V_R = 64V$,
		—	46	69	ns	$T_J = 125^\circ\text{C}$ $I_F = 75\text{A}$
Q_{rr}	Reverse Recovery Charge	—	65	98	nC	$T_J = 25^\circ\text{C}$ $dI/dt = 100\mu\text{A}/\mu\text{s}$ ⑤
		—	86	130	ns	$T_J = 125^\circ\text{C}$
I_{RRM}	Reverse Recovery Current	—	2.8	—	A	$T_J = 25^\circ\text{C}$
t_{on}	Forward Turn-On Time	Intrinsic turn-on time is negligible (turn-on is dominated by LS+LD)				

Notes:

- ① Calculated continuous current based on maximum allowable junction temperature. Package limitation current is 75A.
- ② Repetitive rating; pulse width limited by max. junction temperature.
- ③ Limited by $T_{J,\text{max}}$, starting $T_J = 25^\circ\text{C}$, $L = 0.096\text{mH}$, $R_G = 25\Omega$, $I_{AS} = 75\text{A}$, $V_{GS} = 10V$. Part not recommended for use above this value.
- ④ $I_{SD} \leq 75\text{A}$, $dI/dt \leq 530\text{A}/\mu\text{s}$, $V_{DD} \leq V_{(\text{BR})DSS}$, $T_J \leq 175^\circ\text{C}$.
- ⑤ Pulse width $\leq 400\mu\text{s}$; duty cycle $\leq 2\%$.
- ⑥ C_{oss} eff. (TR) is a fixed capacitance that gives the same charging time as C_{oss} while V_{DS} is rising from 0 to 80% V_{DSS} .
- ⑦ C_{oss} eff. (ER) is a fixed capacitance that gives the same energy as C_{oss} while V_{DS} is rising from 0 to 80% V_{DSS} .
- ⑧ When mounted on 1" square PCB (FR-4 or G-10 Material). For recommended footprint and soldering techniques refer to application note #AN-994. R_G is measured at T_J approximately 90°C .

International
IR Rectifier

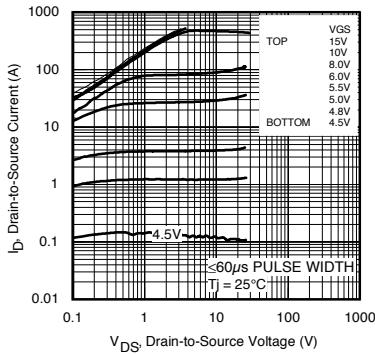


Fig 1. Typical Output Characteristics

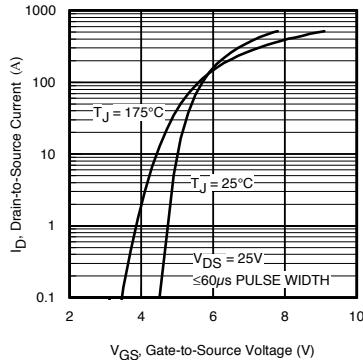


Fig 3. Typical Transfer Characteristics

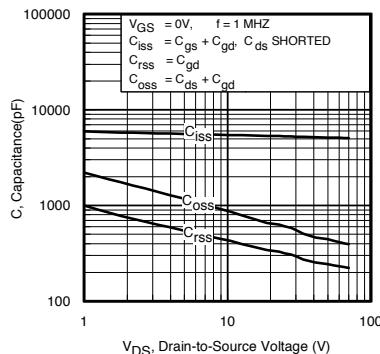


Fig 5. Typical Capacitance vs. Drain-to-Source Voltage
www.irf.com

IRFB3307/IRFS3307/IRFSL3307

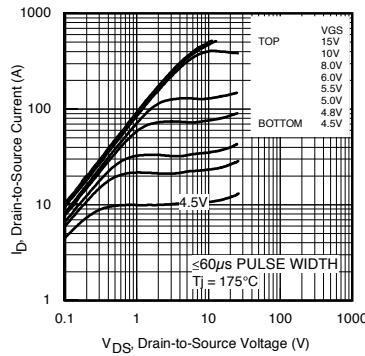


Fig 2. Typical Output Characteristics

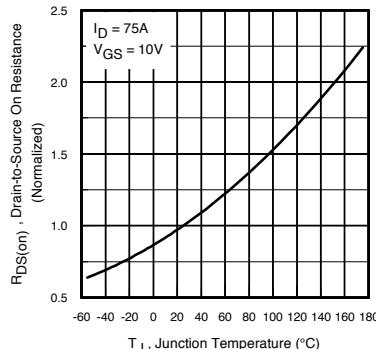


Fig 4. Normalized On-Resistance vs. Temperature

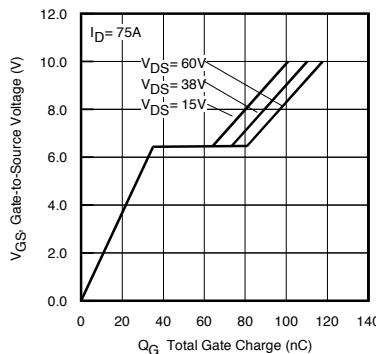


Fig 6. Typical Gate Charge vs. Gate-to-Source Voltage

IRFB3307/IRFS3307/IRFSL3307

International
IR Rectifier

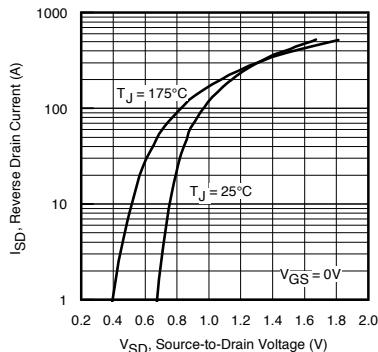


Fig 7. Typical Source-Drain Diode Forward Voltage

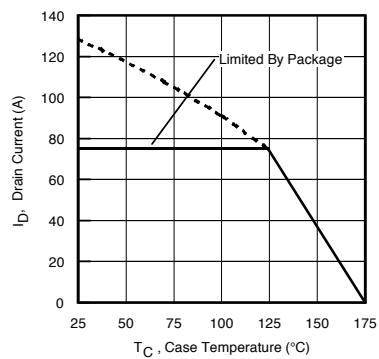


Fig 9. Maximum Drain Current vs. Case Temperature

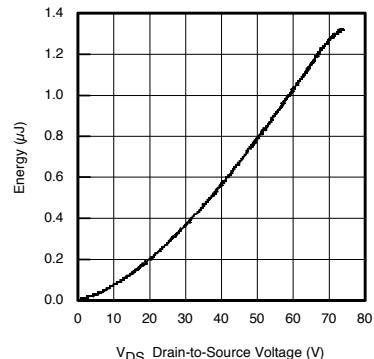


Fig 11. Typical C_{oss} Stored Energy

4

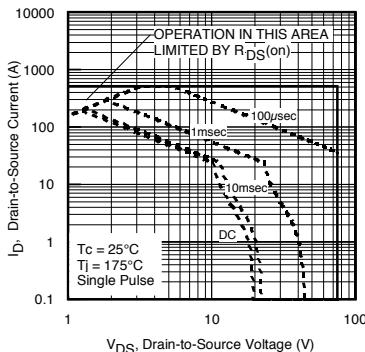


Fig 8. Maximum Safe Operating Area

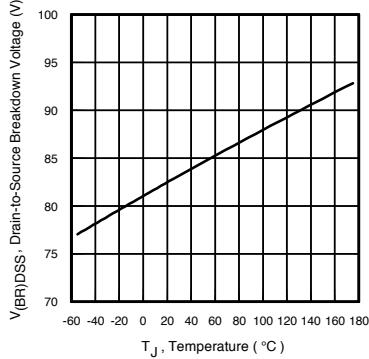


Fig 10. Drain-to-Source Breakdown Voltage

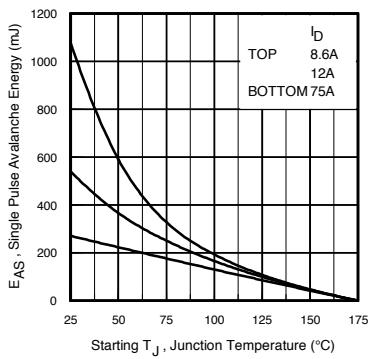


Fig 12. Maximum Avalanche Energy vs. Drain Current

www.irf.com

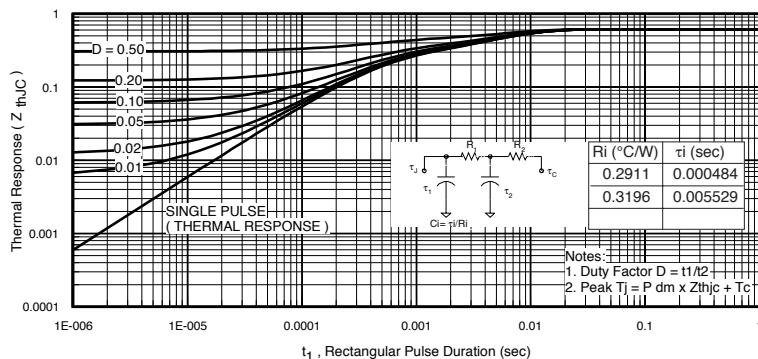


Fig 13. Maximum Effective Transient Thermal Impedance, Junction-to-Case

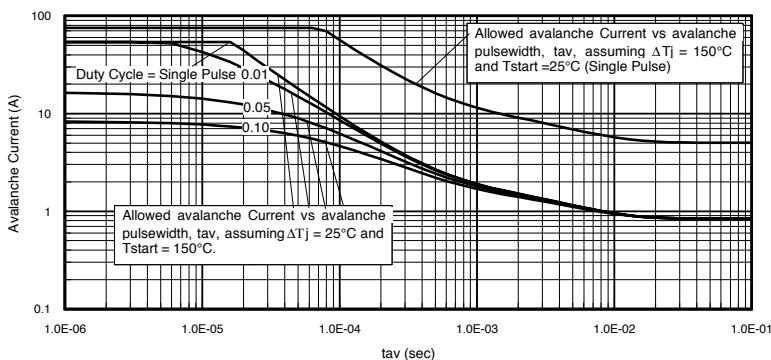


Fig 14. Typical Avalanche Current vs.Pulsewidth

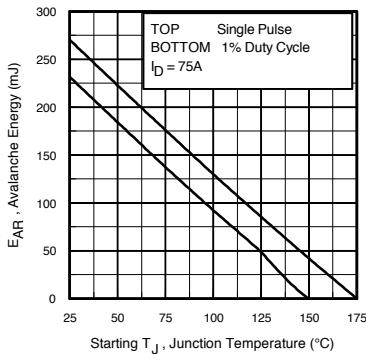


Fig 15. Maximum Avalanche Energy vs. Temperature

Notes on Repetitive Avalanche Curves , Figures 14, 15:
(For further info, see AN-1005 at www.irf.com)

1. Avalanche failures assumption:
Purely a thermal phenomenon and failure occurs at a temperature far in excess of $T_{j\max}$. This is validated for every part type.
 2. Safe operation in Avalanche is allowed as long as neither $T_{j\max}$ nor I_{av} (max) is exceeded.
 3. Equation below based on circuit and waveforms shown in Figures 16a, 16b.
 4. $P_{D(\text{ave})}$ = Average power dissipation per single avalanche pulse.
 5. BV = Rated breakdown voltage (1.3 factor accounts for voltage increase during avalanche).
 6. I_{av} = Allowable avalanche current.
 7. ΔT = Allowable rise in junction temperature, not to exceed $T_{j\max}$ (assumed as 25°C in Figure 14, 15).
- t_{av} = Average time in avalanche.
 D = Duty cycle in avalanche = $t_{av} \cdot f$
 $Z_{\text{ThJC}}(D, t_{av})$ = Transient thermal resistance, see Figures 13)

$$P_{D(\text{ave})} = 1/2 \cdot (1.3 \cdot BV \cdot I_{av}) = \Delta T / Z_{\text{ThJC}}$$

$$I_{av} = 2 \cdot \Delta T / [1.3 \cdot BV \cdot Z_{\text{ThJC}}]$$

$$E_{AS(AR)} = P_{D(\text{ave})} \cdot t_{av}$$

IRFB3307/IRFS3307/IRFSL3307

International
IR Rectifier

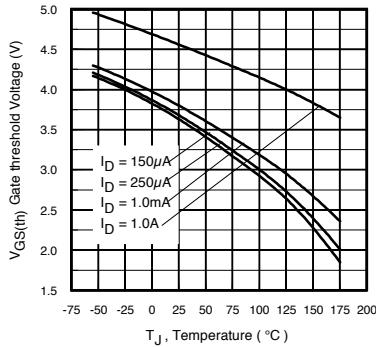


Fig. 16. Threshold Voltage vs. Temperature

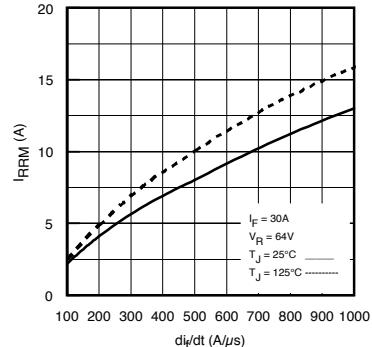


Fig. 17 - Typical Recovery Current vs. di_F/dt

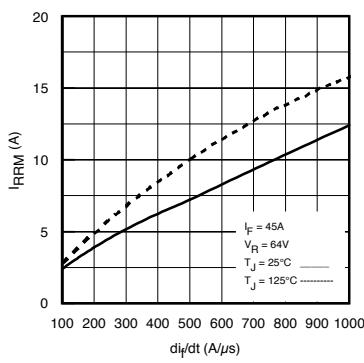


Fig. 18 - Typical Recovery Current vs. di_F/dt

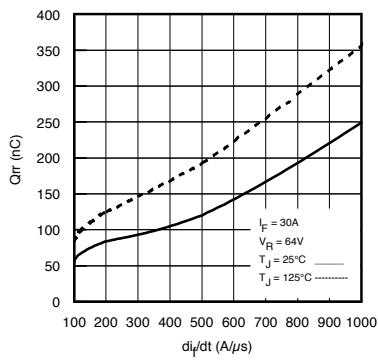


Fig. 19 - Typical Stored Charge vs. di_F/dt

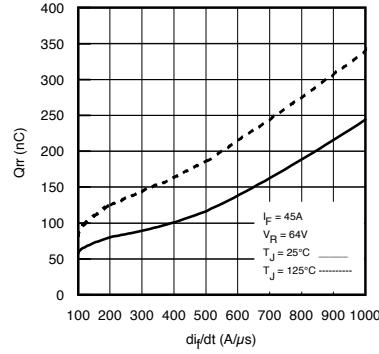
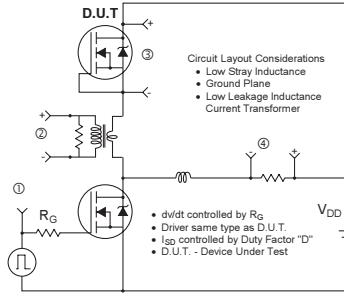


Fig. 20 - Typical Stored Charge vs. di_F/dt

International
IR Rectifier



IRFB3307/IRFS3307/IRFSL3307

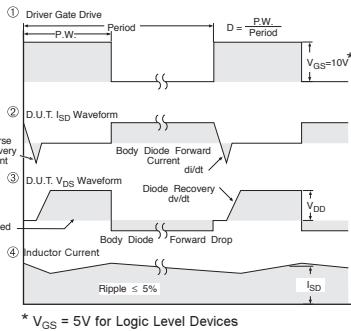


Fig 20. Peak Diode Recovery dv/dt Test Circuit for N-Channel HEXFET® Power MOSFETs

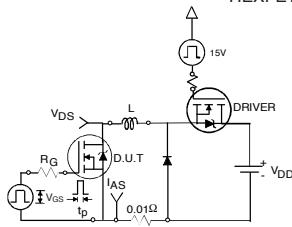


Fig 21a. Unclamped Inductive Test Circuit

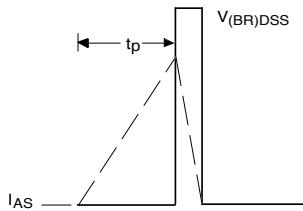


Fig 21b. Unclamped Inductive Waveforms

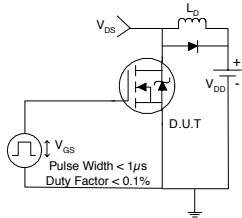


Fig 22a. Switching Time Test Circuit

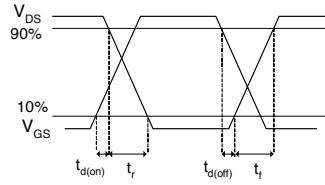


Fig 22b. Switching Time Waveforms

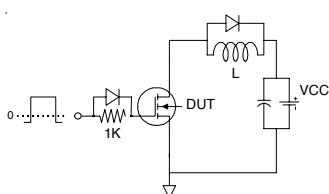


Fig 23a. Gate Charge Test Circuit
www.irf.com

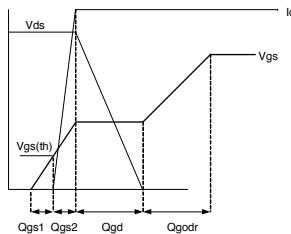


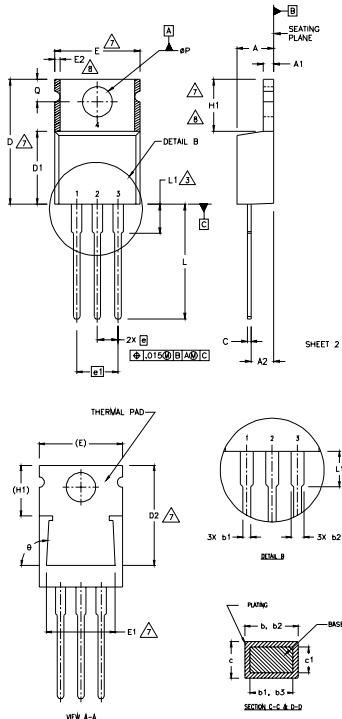
Fig 23b. Gate Charge Waveform

IRFB3307/IRFS3307/IRFSL3307

International
IR Rectifier

TO-220AB Package Outline

Dimensions are shown in millimeters (inches)



- NOTES:**
1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M - 1994.
 2. DIMENSIONS ARE SHOWN IN INCHES (MILLIMETERS).
 3. LEAD DIMENSION AND FINISH UNCONTROLLED IN L1.
 4. DIMENSIONS E & H1 DO NOT INCLUDE WIRE FLAT OR VOID FLASH. SHOWS NOT EXCEED .005" (.127MM) ON SIZE. THESE DIMENSIONS ARE MEASURED AT THE OUTERMOST EXTREMES OF THE PLASTIC BODY.
 5. DIMENSION D1 & C1 APPLY TO BASE METAL ONLY.
 6. CONTROLLING DIMENSION : INCHES.
 7. THERMAL PAD CONTOUR OPTIONS WITHIN DIMENSIONS E,H1,D2 & E1
 8. DIMENSION E2 X H1 DEFINE A ZONE WHERE STAMPING AND SINGULATION IRRREGULARITIES ARE ALLOWED.

LEAD ASSIGNMENTS

- GATE:
1- GATE
2- DRAIN
3- SOURCE

- COLLECTOR:
1- GATE
2- COLLECTOR
3- EMITTER

- DOORS:
1- ANODE/OPEN
2- CATHODE
3- PACT

SYMBOL	DIMENSIONS		NOTES	
	MILLIMETERS	INCHES		
	MIN.	MAX.	MIN.	MAX.
A	.356	.482	.140	.190
A1	.051	.140	.020	.055
A2	2.04	2.92	.080	.115
b	.038	.101	.015	.040
b1	.038	.096	.015	.038
b2	.15	.177	.045	.070
b3	.15	.173	.045	.068
c	.036	.061	.014	.024
c1	.036	.056	.014	.022
D	14.22	16.51	.560	.660
D1	8.38	9.02	.330	.355
D2	12.19	12.88	.480	.507
E	9.66	10.66	.380	.420
E1	8.38	8.89	.330	.350
e	2.54 .050	3.00 .050		
e1	5.08	5.08		
H1	5.85	6.55	.230	.270
L	12.70	14.73	.500	.580
L1			.335	.250
gP	3.54	4.08	.139	.161
o	2.54	3.42	.100	.135
θ	90° - 93°		90° - 93°	

TO-220AB Part Marking Information

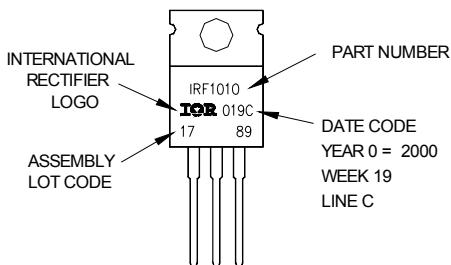
EXAMPLE: THIS IS AN IRF1010

LOT CODE 1789

ASSEMBLED ON WW 19, 2000

IN THE ASSEMBLY LINE "C"

Note: "P" in assembly line position indicates "Lead - Free"

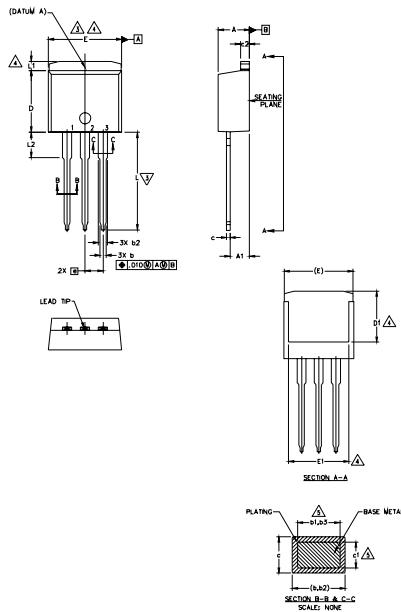


TO-220AB packages are not recommended for Surface Mount Application.

International
IR Rectifier

TO-262 Package Outline

Dimensions are shown in millimeters (inches)



IRFB3307/IRFS3307/IRFSL3307

NOTES:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M-1994
2. DIMENSIONS ARE SHOWN IN MILLIMETERS [INCHES]
3. DIMENSION D & E DO NOT INCLUDE MOLD FLASH. MOLD FLASH SHALL NOT EXCEED 0.127 (.005") PER SIDE. THESE DIMENSIONS ARE MEASURED AT THE OUTMOST EXTREMES OF THE PLASTIC BODY.
4. THERMAL PAD CONTOUR OPTIONAL WITHIN DIMENSION E, L1, D1 & E1.
5. DIMENSION b1 AND c1 APPLY TO BASE METAL ONLY.
6. CONTROLLING DIMENSION: INCH.
7. OUTLINE CONFORM TO JEDEC TO-262 EXCEPT A1(max.), b(min.) AND D1(min.), WHERE DIMENSIONS DERIVED THE ACTUAL PACKAGE OUTLINE.

S Y M B O	DIMENSIONS			N O T E S
	MILLIMETERS	INCHES		
L	MIN.	MAX.	MIN.	MAX.
A	4.06	4.83	.160	.190
A1	2.03	3.02	.080	.119
b	0.51	0.99	.020	.039
b1	0.51	0.89	.020	.035
b2	1.14	1.78	.045	.070
b3	1.14	1.73	.045	.068
c	0.38	0.74	.015	.029
c1	0.38	0.58	.015	.023
c2	1.14	1.65	.045	.065
D	8.38	9.65	.330	.380
D1	6.86	-	.270	-
E	9.65	10.67	.380	.420
E1	6.22	-	.245	-
e	2.54	BSC	.100	BSC
L	13.46	14.10	.530	.555
L1	-	1.65	-	.065
L2	3.56	3.71	.140	.146

LEAD ASSIGNMENTS

HEXFET

1. GATE
2. DRAIN
3. SOURCE
4. DRAIN

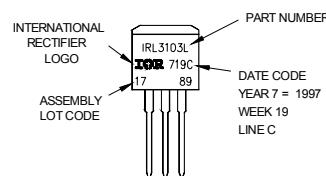
IGBTs, CapACK

1. GATE
2. COLLECTOR
3. Emitter
4. COLLECTOR

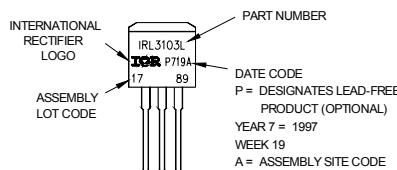
TO-262 Part Marking Information

EXAMPLE: THIS IS AN IRL3103L
LOT CODE 1789
ASSEMBLED ON WW 19, 1997
IN THE ASSEMBLY LINE "C"

Note: "P" in assembly line position
indicates "Lead - Free"



OR

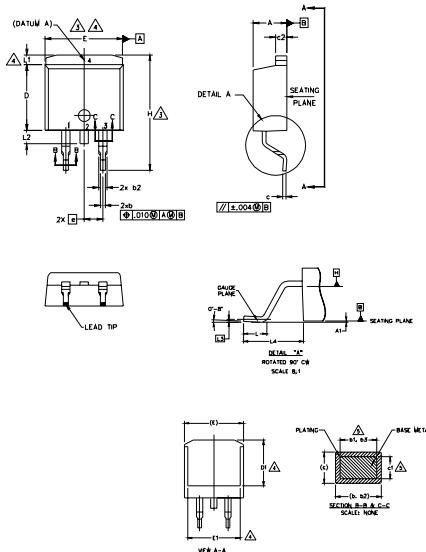


IRFB3307/IRFS3307/IRFSL3307

International
IR Rectifier

D²Pak (TO-263AB) Package Outline

Dimensions are shown in millimeters (inches)



NOTES:
1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M-1994
2. DIMENSIONS ARE SHOWN IN MILLIMETERS [INCHES]

△ DIMENSION D & E DO NOT INCLUDE MOLD FLASH. MOLD FLASH SHALL NOT EXCEED 0.127 (.005") PER SIDE. THESE DIMENSIONS ARE MEASURED AT THE OUTMOST EXTREMES OF THE PLASTIC BODY AT DATUM H.

△ THERMAL PAD CONTOUR OPTIONAL WITHIN DIMENSION E, L1, D1 & E1.

△ DIMENSION B1 & C1 APPLY TO BASE METAL ONLY.

6. DATUM A & B TO BE DETERMINED AT DATUM PLANE H.

7. CONTROLLING DIMENSION: INCH.

8. OUTLINE CONFORMS TO JEDEC OUTLINE TO-263AB.

S I	DIMENSIONS				N T S
	M MILLIMETERS	I INCHES	M MIN.	M MAX.	
A	4.06	.160	.160	.190	
A1	0.00	.0254	.000	.010	
b	0.51	.099	.020	.039	
b1	0.51	.099	.020	.035	5
b2	1.14	.178	.045	.070	
b3	1.14	.173	.045	.068	5
c	0.38	.074	.015	.029	
c1	0.38	.058	.015	.023	5
c2	1.14	.165	.045	.065	
D	8.38	.965	.330	.380	3
D1	6.86	—	.270	—	4
E	9.65	10.67	.380	.420	3,4
E1	6.22	—	.245	—	4
e	2.54 BSC	—	.100 BSC	—	
H	14.61	15.88	.575	.625	
L	1.78	2.79	.070	.110	
L1	—	1.65	—	.066	4
L2	1.27	1.78	—	.070	
L3	0.25 BSC	—	.010 BSC	—	
L4	4.78	5.28	.188	.208	

LEAD ASSIGNMENTS

HEXFET
1. - GATE
2. 4. - DRAIN
3. - SOURCE

IGBTs, CoPACK
1. - GATE
2. 4. - COLLECTOR
3. - Emitter

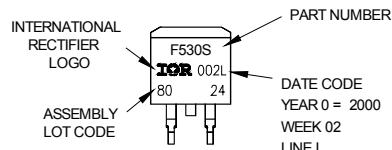
DIODES
1. - ANODE *
2. 4. - CATHODE
3. - ANODE

* PART DEPENDENT.

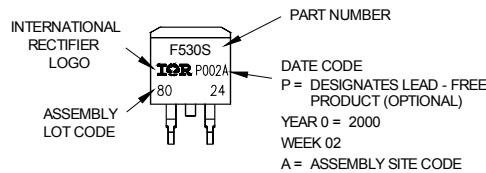
D²Pak (TO-263AB) Part Marking Information

EXAMPLE: THIS IS AN IRF530S WITH
LOT CODE 8024
ASSEMBLED ON WW 02, 2000
IN THE ASSEMBLY LINE "L"

Note: "P" in assembly line position
indicates "Lead - Free"



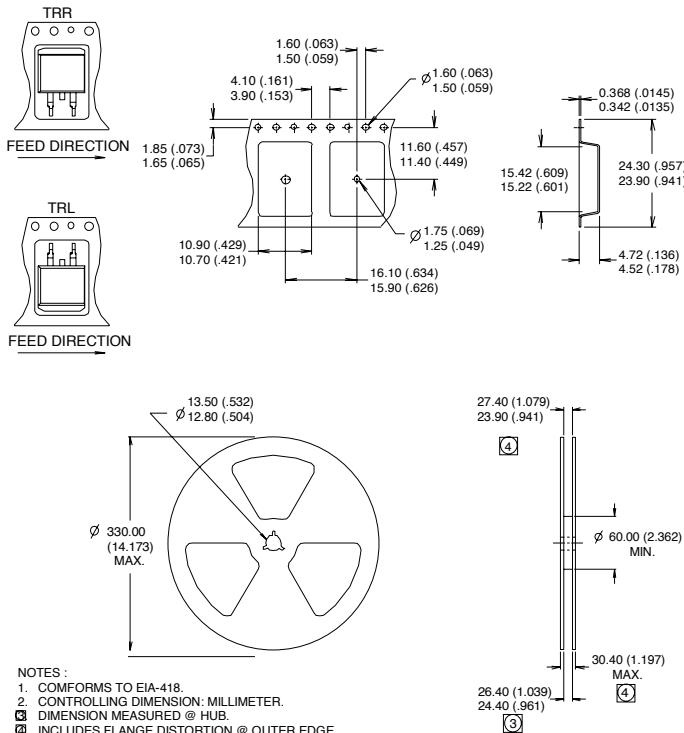
OR



International
IR Rectifier

IRFB3307/IRFS3307/IRFSL3307

D²Pak (TO-263AB) Tape & Reel Information



Data and specifications subject to change without notice.
This product has been designed and qualified for the Industrial market.
Qualification Standards can be found on IR's Web site.

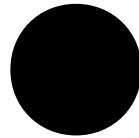
International
IR Rectifier

IR WORLD HEADQUARTERS: 233 Kansas St., El Segundo, California 90245, USA Tel: (310) 252-7105
TAC Fax: (310) 252-7903
Visit us at www.irf.com for sales contact information. 01/06

Document Information

Project reference

317947, FP7-ICT-2011-8



Context

Deliverable 2.1, workpackage 2

Document title

201309_Arduino_Sands_Deliverable_2-1_Relay

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

1. Sep. 2013

Table of contents

Document Information.....	1
Table of contents.....	2
Summary.....	3
Purpose within SandS.....	3
SandS Relay module.....	4
AVR core.....	4
Power the board.....	4
Hardware.....	5
Board dimensions.....	6
Mounting considerations.....	6
Schematic.....	6
Board file.....	6
Pin layout.....	6
Firmware.....	6
Licenses.....	7
Hardware.....	7
Firmware.....	7
Appendix 1 - Schematics.....	8

Summary

This document describes the main characteristics of the SandS Relay module, an open source hardware - Arduino compatible board, exposing a 5V serial port, and a SandS-enhanced I2C port. This board runs a series of commands to control its on-board relay.

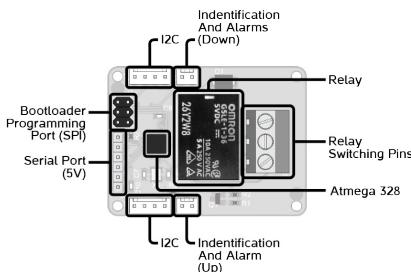


Image 1: block description, Relay module

The use scenario for this board is to be switching on and off the power on any appliance. The voltage allowance of the relay is 250VAC.

The board runs from the power provided by the I2C port. It can be reprogrammed from the Arduino IDE using a USB-Serial cable and the SandS special library. It comes with a pre-loaded firmware and runs a modified version of the I2C protocol that allows automatic identification. In other words, it is possible to dynamically assign addresses to the board.

Even if the board has been specifically designed for the SandS project, the pins exposed directly from the processor (the ATmega328) can be addressed as Arduino digital pins.

Purpose within SandS

The purpose of this module is to switch different electric components like AC motors, lamps, big heaters, etc. The main difference between this Relay module and the Mosfet module is the time granularity, as the Relay module switching capability happens at a much slower frequency than with the Mosfet module.

SandS Relay module

The SandS relay module is a microcontroller board operating as a I2C peripheral to the SandS motherboard. It is made of two blocks:

- a microcontroller (the ATmega328): running at 16MHz it listens to the I2C port and executes the different operations on the module
- a relay used to switch the power on any external devices

It comes with two connections for the SandS enhanced I2C (two-wire interface + IDENT + ALARM), one 5V RS-232 port, and the SPI programming port to burn the processor's bootloader. See Image 1 for a block diagram of the board.

AVR core

The AVR core chosen for the SandS relay module is the ATmega328, which main characteristics are described in Table 1:

Processor	ATmega328
Flash Memory	32 KB of which 512 B used by bootloader
SRAM	2 KB
EEPROM	1 KB
Operating Voltage	5 V
Hardware Serial ports	1
Hardware SPI ports	1
Hardware I2C ports	1

Table 1: main characteristics of the AVR core

This processor can be programmed as an Arduino UNO board directly from the standard Arduino IDE, there is more information how to program the board on the *Getting Started* guide to the module.

The ATmega328 does not have an internal USB peripheral, therefore, the SandS relay module needs using a USB-Serial cable, as a way to achieve USB communication.

Power the board

Input Voltage (recommended)	5 V through the I2C connector
Current (min)	50 mA

Table 2: powering the board

All the SandS modules are powered via the I2C cable. However, during reprogramming operations, it is possible to power the modules through the 5V serial connector.

Hardware

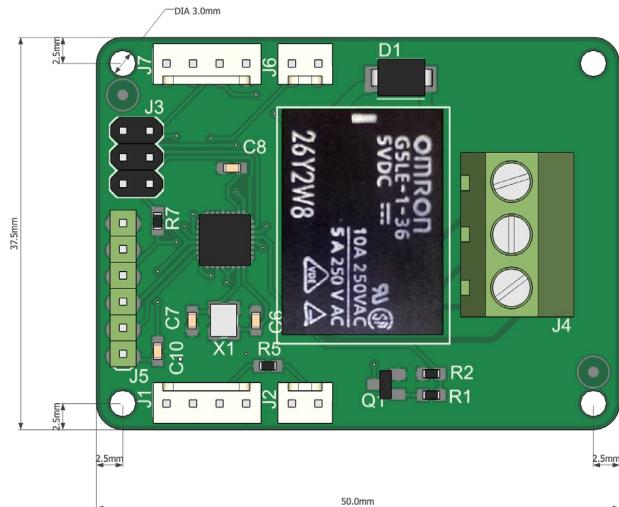


Image 2: Top view, Sands relay module v0002

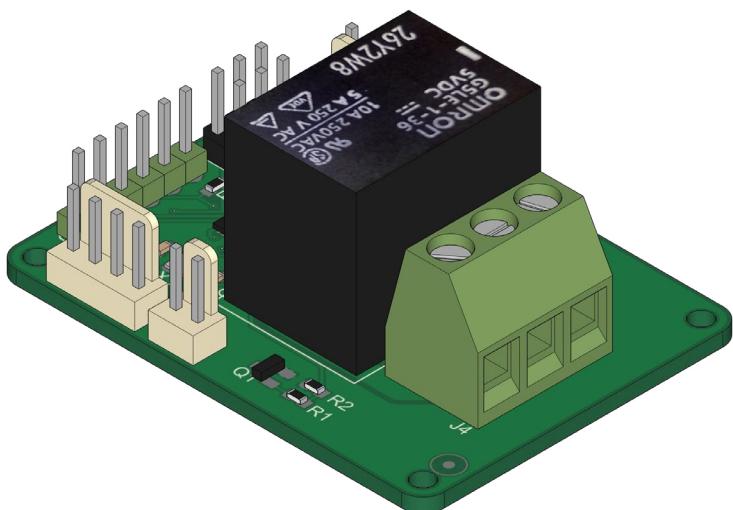


Image 3: Isometric view of the SandS relay module v0002

Board dimensions

The board has been designed to easily accommodate the parts in a 2-layer design in the size of 37,5x50mm. It comes with 3mm holes at each corner, making it easy to mount it using standard plastic spacers and M3 screws.

Check Image 2 for further details about measurements and location of the holes on the board.

Image 3 gives you an idea of the proportions of the different components on the board.

Mounting considerations

It is recommended to mount the board in a plastic box. The relay module is meant to be switching high AC voltage, a metallic box could provoke a shortcircuit that would damage the board and be potentially harmful.

The module will be connected as part of a I2C bus. This means that the board will use standard board-to-cable connectors both for the uplink and the downlink of the I2C and therefore it is recommended leaving space for those cables to exit the box. Alternatively it is possible to mount differen types of connectors on the plastic housing and simply extend the board with those.

For safety purposes, the relay connects through a screw connector. If you were in the need to use a different type of connector, you should fix it to the board's box and plug it in to the screw connector.

Schematic

The board's schematic shows all the components on it. You can check all the schematics at Appendix 1, at the end of this document.

The schematics come as a single document showing:

- a relay being switched via a bipolar transistor
- Microcontroller: is the ATmega328, with a 16Mhz crystal
- the SPI connector for reprogramming the processor's bootloader
- the I2C uplink and downlink connectors
- the **IDENT** and **ALARM** connectors
- the serial port connection for reprogramming the device and debugging it's firmware

Board file

The board file is a 2-layered design with the components separated strategically to avoid potential interferences. Half of the board is reserved for the actual relay, since it is meant to switch up to 250 VAC, while the other half is

dedicated to the AVR core and all the connectors.

Images 2 and 3 give a pretty good idea of this layout. Both schematic and board file for this design come bundled in the same zipped file as this document and can be modified using the Eagle Cad software.

Pin layout

Image 4 shows the pin layout for the board as well as the description of each one of the pins separately.

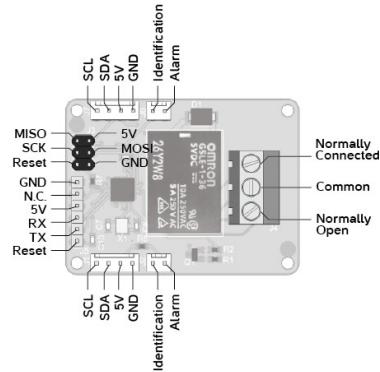


Image 4: pin layout, Relay module

Firmware

The Sands relay board comes with a preprogrammed firmware that will get the processor to receive commands through the I2C protocol and execute them accordingly. The normal operation mode doesn't need of reprogramming the device. All the specific commands for this device are described in depth in the TWI Protocol document that comes bundled with this one.

The firmware is open source and comes installed in the board. It is possible to reprogram it directly from the Arduino IDE. It is also possible to use other tools like command line, or AVR studio to change the firmware in the processor.

Licenses

Hardware

The design of the boards falls under the CERN Open Hardware License 1.2, for more information refer to the following link:

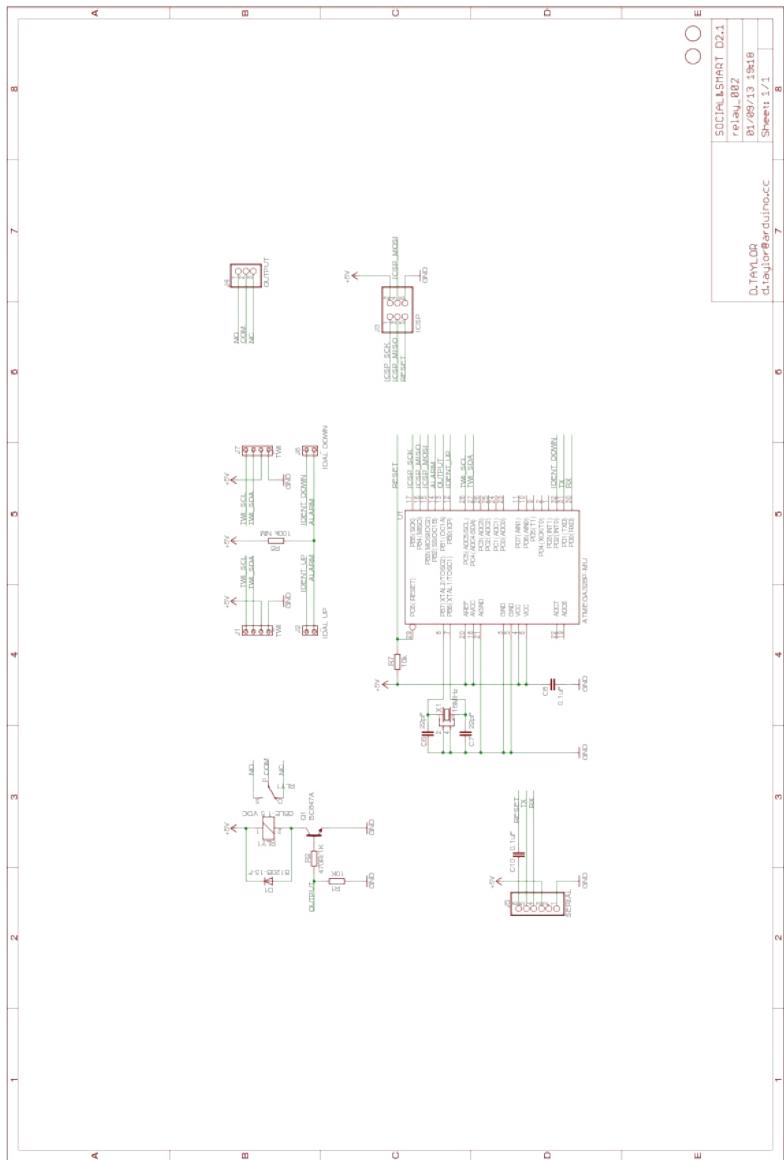
http://ohwr.org/attachments/2388/cern_ohl_v_1_2.txt

Firmware

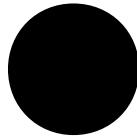
The firmware for the ATmega328 processor used in the SandS Relay board is based on the Arduino core and therefore it is licensed under LGPL, however all the source is available for anybody to use. To read more about this license, refer to:

<https://www.gnu.org/licenses/lgpl.html>

Appendix 1 - Schematics



Document Information



Project reference

317947, FP7-ICT-2011-8

Context

Deliverable 2.1, workpackage 2

Document title

201309_Arduino_Sands_Deliverable_2-1_Sensor

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

1. Sep. 2013

Table of contents

Document Information.....	1
Table of contents.....	2
Summary.....	3
Purpose within SandS.....	3
SandS Input module.....	4
AVR core.....	4
Power the board.....	4
Amplifier.....	4
Hardware.....	5
Board dimensions.....	6
Mounting considerations.....	6
Schematic.....	6
Board file.....	6
Pin layout.....	6
Firmware.....	6
Licenses.....	7
Hardware.....	7
Firmware.....	7
Appendix 1 – Schematics.....	8
Appendix 2 - Sensor Datasheet.....	10

Summary

This document describes the main characteristics of the SandS Input module, an open source hardware - Arduino compatible board, exposing a 5V serial port, and a SandS-enhanced I2C port. This board runs a series of commands to control its on-board amplifier in order to make readings of different types of sensors connected to it.

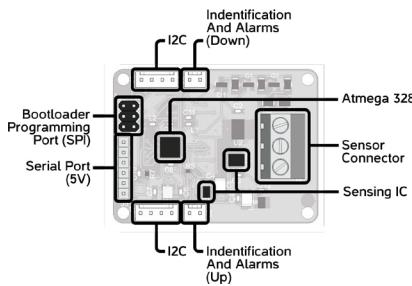


Image 1: block description, input module

The use scenario for this board is to be reading voltages coming from different types of sensors offering a simple way to control the reading range through a digital interface to the sensors.

The voltage allowance of the sensor is up to 24 VDC, it is possible to set different voltage ranges using different mosfets controlled from the microprocessor.

There are 14 different levels of gain: 0, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096.

Values are read with a 10-bit depth via the internal ADC peripheral inside the ATmega328.

It is possible configuring the board to make either analog or digital readings by issuing different I2C commands.

The board runs from the power provided by the I2C port. It can be reprogrammed from the Arduino IDE using a USB-Serial cable and the SandS special library. It comes with a pre-loaded firmware and runs a modified version of the I2C protocol that allows automatic identification. In other words, it is possible to dynamically assign addresses to the board.

Even if the board has been specifically designed for the SandS project, the pins exposed directly from the processor (the ATmega328) can be addressed as Arduino digital pins.

Purpose within SandS

The purpose of this module is to read different sensors that are used in different appliances like NTCs, PTCs, varistors, etc. These devices are common use in state of

SandS Input module

The SandS input module is a microcontroller board operating as a I2C peripheral to the SandS motherboard. It is made of two blocks:

- a microcontroller (the ATmega328): running at 16MHz it listens to the I2C port and executes the different operations on the module
- a digital amplifier used to improve the readability of an input signal coming from a multiplicity of sensors and with a maximum gain of 4096

It comes with two connections for the SandS enhanced I2C (two-wire interface + IDENT + ALARM), one 5V RS-232 port, and the SPI programming port to burn the processor's bootloader. See Image 1 for a block diagram of the board.

AVR core

The AVR core chosen for the SandS input module is the ATmega328, which main characteristics are described in Table 1:

Processor	ATmega328
Flash Memory	32 KB of which 512 B used by bootloader
SRAM	2 KB
EEPROM	1 KB
Operating Voltage	5 V
Hardware Serial ports	1
Hardware SPI ports	1
Hardware I2C ports	1

Table 1: main characteristics of the AVR core

This processor can be programmed as an Arduino UNO board directly from the standard Arduino IDE, there is more information how to program the board on the *Getting Started* guide to the module.

The ATmega328 does not have an internal USB peripheral, therefore, the SandS input module needs using a USB-Serial cable, as a way to achieve USB communication.

Power the board

Input Voltage (recommended)	5 V through the I2C connector
Current (min)	50 mA

Table 2: powering the board

All the SandS modules are powered via the I2C cable. However, during reprogramming operations, it is possible to power the modules through the 5V serial connector.

Amplifier

Gain (max)	4096, 14 levels
Gain accuracy	0,1%
Voltage (max)	24 V
Current (max)	10 mA
Current (min)	-10 mA
Offset Volt. (max)	10µV

Table 3: amplifier characteristics

The amplifier chosen for this circuit is the LTC6915 - Zero Drift, Precision Instrumentation Amplifier with Digitally Programmable Gain from Linear instruments. It is a device controlled via a serial port interface, however the readings are made by the ADC (Analog to Digital Converter) inside the ATmega328, which has 10 bits of depth.

For further information about the amplifier, it is recommended checking the datasheet for the amplification IC, inside the Appendix 2.

Hardware

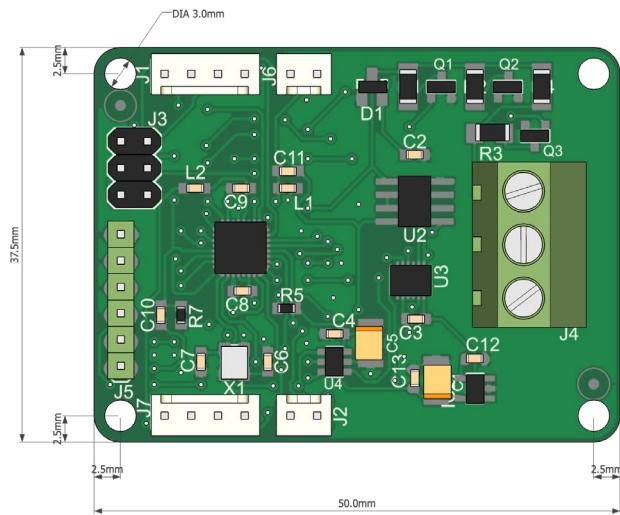


Image 2: Top view, SandS input module v0002

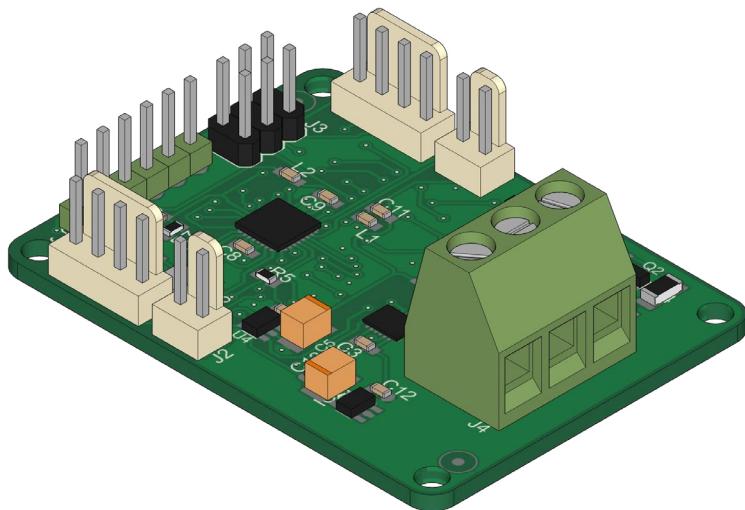


Image 3: Isometric view of the SandS input module v0002

Board dimensions

The board has been designed to easily accommodate the parts in a 2-layer design in the size of 37,5x50mm. It comes with 3mm holes at each corner, making it easy to mount it using standard plastic spacers and M3 screws.

Check Image 2 for further details about measurements and location of the holes on the board.

Image 3 gives you an idea of the proportions of the different components on the board.

Mounting considerations

It is recommended to mount the board in a plastic box.

The module will be connected as part of a I2C bus. This means that the board will use standard board-to-cable connectors both for the uplink and the downlink of the I2C and therefore it is recommended leaving space for those cables to exit the box. Alternatively it is possible to mount different types of connectors on the plastic housing and simply extend the board with those.

The input module is meant to be reading voltages coming from different types of sensors: capacitive, resistive, etc. The sensing IC is accessible through a screw connector. If you were in the need to use a different type of connector, you should fix it to the board's box and plug it in to the screw connector.

Schematic

The board's schematic shows all the components on it. You can check all the schematics at Appendix 1, at the end of this document.

The schematics come as a single document showing:

- a digital amplifier controlled by the microprocessors
- a couple of operational amplifiers used to ensure signal integrity and offer digital readings from sensors
- a series of mosfets used to determine the possible input range
- Microcontroller: is the ATmega328, with a 16Mhz crystal
- the SPI connector for reprogramming the processor's bootloader
- the I2C uplink and downlink connectors
- the IDENT and ALARM connectors
- the serial port connection for reprogramming the device and debugging its firmware

Board file

The board file is a 2-layered design with the components separated strategically to avoid potential interferences. Half of the board is reserved for the sensing circuitry, while the other half is dedicated to the AVR core and all the connectors.

Images 2 and 3 give a pretty good idea of this layout. Both schematic and board file for this design come bundled in the same zipped file as this document and can be modified using the Eagle Cad software.

Pin layout

Image 4 shows the pin layout for the board as well as the description of each one of the pins separately.

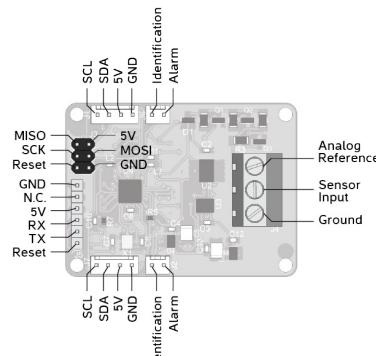


Image 4: pin layout, Input module

Firmware

The SandS input board comes with a preprogrammed firmware that will get the processor to receive commands through the I2C protocol and execute them accordingly. The normal operation mode doesn't need of reprogramming the device. All the specific commands for this device are described in depth in the TWI Protocol document that comes bundled with this one.

The firmware is open source and comes installed in the board. It is possible to reprogram it directly from the Arduino IDE. It is also possible to use other tools like command line, or AVR studio to change the firmware in the processor.

Licenses

Hardware

The design of the boards falls under the CERN Open Hardware License 1.2, for more information refer to the following link:

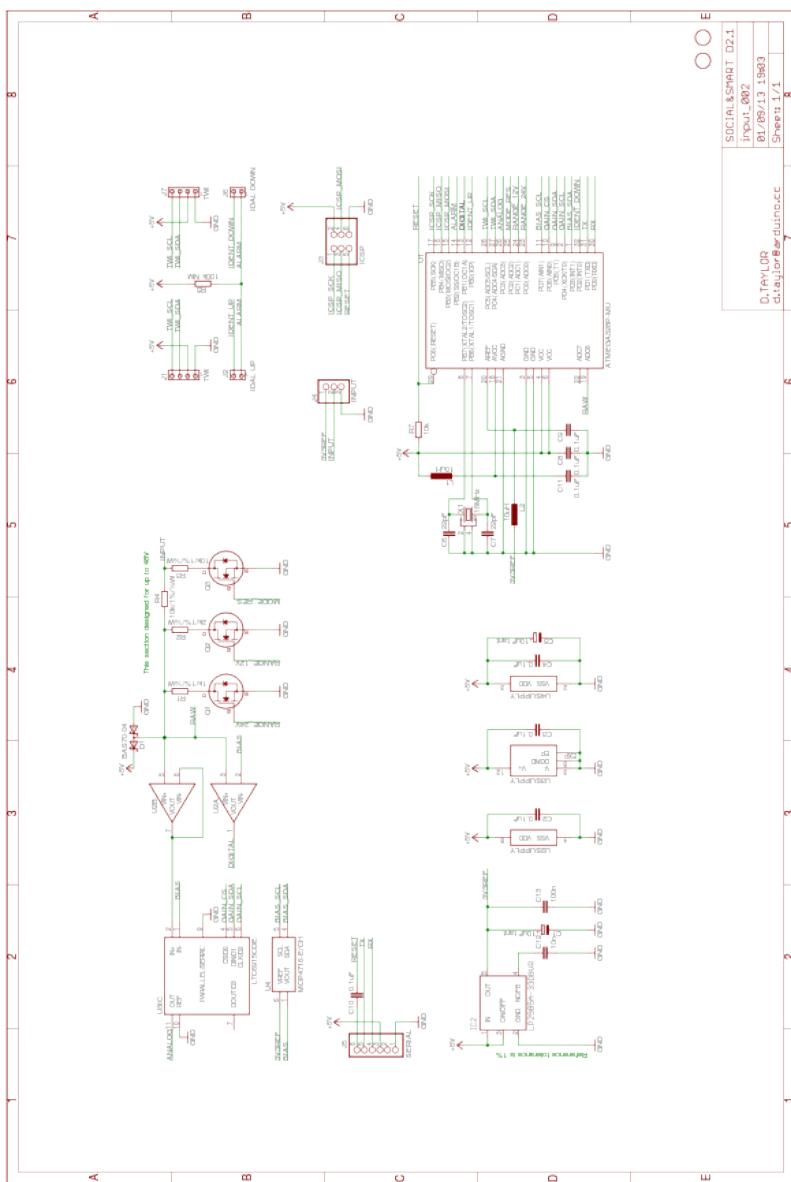
http://ohwr.org/attachments/2388/cern_ohl_v_1_2.txt

Firmware

The firmware for the ATmega328 processor used in the SandS Input board is based on the Arduino core and therefore it is licensed under LGPL, however all the source is available for anybody to use. To read more about this license, refer to:

<https://www.gnu.org/licenses/lgpl.html>

Appendix 1 – Schematics



SOCIAL&SMART D2.1
INPUT 1602
81.08/1.3.1983
Sheet 1/1
8

Appendix 2 - Sensor Datasheet



LTC6915

Zero Drift, Precision
Instrumentation Amplifier with
Digitally Programmable Gain

FEATURES

- 14 Levels of Programmable Gain
- 125dB CMRR Independent of Gain
- Gain Accuracy 0.1% (Typ)
- Maximum Offset Voltage of 10 μ V
- Maximum Offset Voltage Drift: 50nV/ $^{\circ}$ C
- Rail-to-Rail Input and Output
- Parallel or Serial (SPI) Interface for Gain Setting
- Supply Operation: 2.7V to \pm 5.5V
- Typical Noise: 2.5 μ V_{P-P} (0.01Hz to 10Hz)
- 16-Lead SSOP and 12-Lead DFN Packages

APPLICATIONS

- Thermocouple Amplifiers
- Electronic Scales
- Medical Instrumentation
- Strain Gauge Amplifier
- High Resolution Data Acquisition

DESCRIPTION

The LTC®6915 is a precision programmable gain instrumentation amplifier. The gain can be programmed to 0, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, or 4096 through a parallel or serial interface. The CMRR is typically 125dB with a single 5V supply with any programmed gain. The offset is below 10 μ V with a temperature drift of less than 50nV/ $^{\circ}$ C.

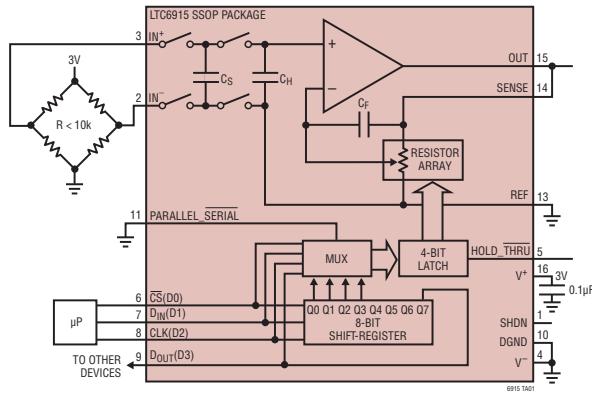
The LTC6915 uses charge balanced sampled data techniques to convert a differential input voltage into a single ended signal that is in turn amplified by a zero-drift operational amplifier.

The differential inputs operate from rail-to-rail and the single-ended output swings from rail-to-rail. The LTC6915 can be used in single power supply applications as low as 2.7V, or with dual \pm 5V supplies. The LTC6915 is available in a 16-lead SSOP package and a 12-lead DFN surface mount package.

LT, LTC, LTM, Linear Technology and the Linear logo are registered trademarks of Linear Technology Corporation. All other trademarks are the property of their respective owners

TYPICAL APPLICATION

Differential Bridge Amplifier with Gain Programmed through the Serial Interface



6915fb

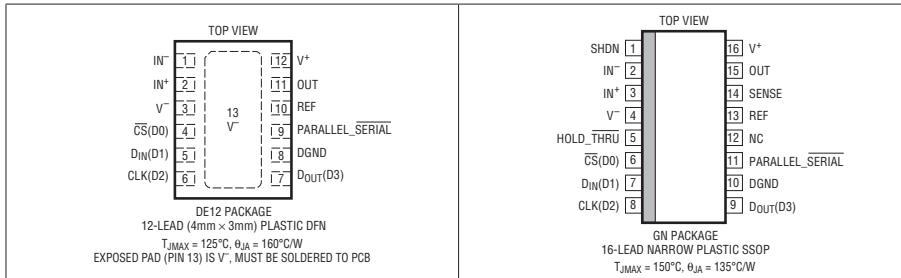
LTC6915

ABSOLUTE MAXIMUM RATINGS

(Note 1)

Total Supply Voltage (V^+ to V^-)	11V	LTC6915I	-40°C to 85°C
Input Current.....	$\pm 10\text{mA}$	LTC6915H	-40°C to 125°C
$ V_{IN^+} - V_{REF} $	5.5V	Junction Temperature	
$ V_{IN^-} - V_{REF} $	5.5V	(GN Package)	150°C
$ V^+ - V_{DGND} $	5.5V	(DFN Package)	125°C
$ V_{DGND} - V^- $	5.5V	Storage Temperature	
Digital Input Voltage.....	V^- to V^+	(GN Package)	-65°C to 150°C
Operating Temperature Range		(DFN Package)	-65°C to 125°C
LTC6915C	-0°C to 70°C	Lead Temperature (Soldering 10 sec).....	300°C

PIN CONFIGURATION



ORDER INFORMATION

LEAD FREE FINISH	TAPE AND REEL	PART MARKING	PACKAGE DESCRIPTION	TEMPERATURE RANGE
LTC6915CDE#PBF	LTC6915CDE#TRPB	6915	12-Lead (4mm × 3mm) Plastic DFN	0°C to 70°C
LTC6915IDE#PBF	LTC6915IDE#TRPB	6915I	12-Lead (4mm × 3mm) Plastic DFN	-40°C to 85°C
LTC6915CGN#PBF	LTC6915CGN#TRPB	6915	16-Lead Narrow Plastic SSOP	0°C to 70°C
LTC6915IGN#PBF	LTC6915IGN#TRPB	6915I	16-Lead Narrow Plastic SSOP	-40°C to 85°C
LTC6915HGN#PBF	LTC6915HGN#TRPB	6915H	16-Lead Narrow Plastic SSOP	-40°C to 125°C
LEAD BASED FINISH	TAPE AND REEL	PART MARKING*	PACKAGE DESCRIPTION	TEMPERATURE RANGE
LTC6915CDE	LTC6915CDE#TR	6915	12-Lead (4mm × 3mm) Plastic DFN	0°C to 70°C
LTC6915IDE	LTC6915IDE#TR	6915I	12-Lead (4mm × 3mm) Plastic DFN	-40°C to 85°C
LTC6915CGN	LTC6915CGN#TR	6915	16-Lead Narrow Plastic SSOP	0°C to 70°C
LTC6915IGN	LTC6915IGN#TR	6915I	16-Lead Narrow Plastic SSOP	-40°C to 85°C
LTC6915HGN	LTC6915HGN#TR	6915H	16-Lead Narrow Plastic SSOP	-40°C to 125°C

Consult LTC Marketing for parts specified with wider operating temperature ranges.

For more information on lead free part marking, go to: <http://www.linear.com/leadfree/>

For more information on tape and reel specifications, go to: <http://www.linear.com/tapeandreel/>

6915fb

ELECTRICAL CHARACTERISTICS The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at $T_A = 25^\circ\text{C}$.

SYMBOL	PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS		
$V^+ = 3\text{V}$, $V^- = 0\text{V}$, $V_{\text{REF}} = 200\text{mV}$								
	Gain Error (Note 2)	$A_V = 1$ ($R_L = 10\text{k}$)	●	-0.075	0	0.075	%	
	Gain Error (Note 2)	$A_V = 2$ to 32 ($R_L = 10\text{k}$)	●	-0.5	0	0.5	%	
	Gain Error (Note 2)	$A_V = 64$ to 1024 ($R_L = 10\text{k}$)	●	-0.6	-0.1	0.6	%	
	Gain Error (Note 2)	$A_V = 2048$, 4096 ($R_L = 10\text{k}$)	●	-1	-0.2	1.0	%	
	Gain Nonlinearity	$A_V = 1$	●	3	15		ppm	
V_{OS}	Input Offset Voltage (Note 3)	$V_{\text{CM}} = 200\text{mV}$		-3	± 10		μV	
	Average Input Offset Drift (Note 3)	$T_A = -40^\circ\text{C}$ to 85°C $T_A = 85^\circ\text{C}$ to 125°C	●		± 50	± 100	$\text{nV}/^\circ\text{C}$ $\text{nV}/^\circ\text{C}$	
I_B	Average Input Bias Current (Note 4)	$V_{\text{CM}} = 1.2\text{V}$	●	5	10		nA	
I_{OS}	Average Input Offset Current (Note 4)	$V_{\text{CM}} = 1.2\text{V}$	●	1.5	3		nA	
v_{en}	Input Noise Voltage	DC to 10Hz		2.5			$\mu\text{V}_{\text{p-p}}$	
$V^+ = 3\text{V}$, $V^- = 0\text{V}$, $V_{\text{REF}} = 200\text{mV}$								
CMRR	Common Mode Rejection Ratio	$A_V = 1024$, $V_{\text{CM}} = 0\text{V}$ to 3V LTC6915C	●	100	119		dB	
		$A_V = 1024$, $V_{\text{CM}} = 0.1\text{V}$ to 2.9V , LTC6915I	●	100	119		dB	
		$A_V = 1024$, $V_{\text{CM}} = 0\text{V}$ to 3V LTC6915I	●	95	119		dB	
		$A_V = 1024$, $V_{\text{CM}} = 0.1\text{V}$ to 2.9V , LTC6915H	●	100	119		dB	
		$A_V = 1024$, $V_{\text{CM}} = 0\text{V}$ to 2.97V , LTC6915H	●	85			dB	
PSRR	Power Supply Rejection Ratio (Note 5)	$V_S = 2.7\text{V}$ to 6V	●	110	116		dB	
		Sourcing $200\mu\text{A}$	●	2.95	2.98		V	
		Sourcing 2mA	●	2.75	2.87		V	
		Sinking $200\mu\text{A}$	●	18	50		mV	
		Sinking 2mA	●	130	300		mV	
Supply Current, Parallel Mode		No Load at OUT, $V_{\text{CM}} = 200\text{mV}$	●	0.88	1.3		mA	
Supply Current, Serial Mode (Note 6)		No Load at OUT, Capacitive Load at Q_{OUT} ($C_L = 15\text{pF}$) Continuous CLK Frequency = 4MHz , $\bar{CS} = \text{LOW}$, Gain Control Code = 0001	●	1.1	1.65		mA	
Supply Current Shutdown		$V_{\text{SHDN}} = 2.7\text{V}$ (Hardware Shutdown) $V_{\text{SHDN}} = 1\text{V}$, Gain Control Code = 0000 (Software Shutdown)	●	1	4		μA	
			●	125	180		μA	
SHDN Input High			●	2.7			V	
SHDN Input Low			●	1			V	
SHDN and HOLD_THRU Input Current (Note 2)			●	5			μA	
Internal Op Amp Gain Bandwidth				200			kHz	
Slew Rate				0.2			$\text{V}/\mu\text{s}$	
Internal Sampling Frequency				3			kHz	
$V^+ = 5\text{V}$, $V^- = 0\text{V}$, $V_{\text{REF}} = 200\text{mV}$								
	Gain Error (Note 2)	$A_V = 1$ ($R_L = 10\text{k}$)	●	-0.075	0	0.075	%	
	Gain Error (Note 2)	$A_V = 2$ to 32 ($R_L = 10\text{k}$)	●	-0.5	0	0.5	%	
	Gain Error (Note 2)	$A_V = 64$ to 1024 ($R_L = 10\text{k}$)	●	-0.6	-0.1	0.6	%	
	Gain Error (Note 2)	$A_V = 2048$, 4096 ($R_L = 10\text{k}$)	●	-1	-0.2	1	%	
	Gain Nonlinearity	$A_V = 1$	●	3	15		ppm	

6915fb

LTC6915

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at $T_A = 25^\circ\text{C}$.

V_{OS}	Input Offset Voltage (Note 3)	$V_{CM} = 200\text{mV}$		-3	± 10	μV
	Average Input Offset Drift (Note 3)	$T_A = -40^\circ\text{C}$ to 85°C $T_A = 85^\circ\text{C}$ to 125°C	●		± 50 ± 100	$\text{nV}/^\circ\text{C}$ $\text{nV}/^\circ\text{C}$
I_{OS}	Average Input Bias Current (Note 4)	$V_{CM} = 1.2\text{V}$	●	5	10	nA
	Average Input Offset Current (Note 4)	$V_{CM} = 1.2\text{V}$	●	1.5	3	nA
CMRR	Common Mode Rejection Ratio	$A_V = 1024$, $V_{CM} = 0\text{V}$ to 5V , LTC6915C $A_V = 1024$, $V_{CM} = 0.1\text{V}$ to 4.9V , LTC6915I $A_V = 1024$, $V_{CM} = 0\text{V}$ to 5V , LTC6915I $A_V = 1024$, $V_{CM} = 0.1\text{V}$ to 4.9V , LTC6915H $A_V = 1024$, $V_{CM} = 0\text{V}$ to 4.97V , LTC6915H	● ● ● ● ●	105 105 95 100 85	125 125 125 125 125	dB dB dB dB dB
PSRR	Power Supply Rejection Ratio (Note 5)	$V_S = 2.7\text{V}$ to 6V	●	110	116	dB
	Output Voltage Swing High	Sourcing $200\mu\text{A}$ Sourcing 2mA	● ●	4.95 4.80	4.99 4.93	V V
	Output Voltage Swing Low	Sinking $200\mu\text{A}$ Sinking 2mA	● ●	17 120	50 300	mV mV

$V^+ = 5\text{V}$, $V^- = 0\text{V}$, $V_{REF} = 200\text{mV}$

	Supply Current, Parallel Mode	No Load at OUT, $V_{CM} = 200\text{mV}$	●	0.95	1.48	mA
	Supply Current, Serial Mode (Note 6)	No Load at OUT, Capacitive Load at D_{OUT} ($C_L = 15\text{pF}$; Continuous CLK Frequency = 4MHz , CS = LOW, Gain Control Code = 0001)	●	1.4	2	mA
	Supply Current, Shutdown	$V_{SHDN} = 4.5\text{V}$ (Hardware Shutdown) $V_{SHDN} = 1\text{V}$, Gain Control Code = 0000 (Software Shutdown)	● ●	2 135	10 200	μA μA
	SHDN Input High		●	4.5		V
	SHDN Input Low		●		1	V
	SHDN and HOLD_THRU Input Current (Note 2)		●		5	μA
	Internal Op Amp Gain Bandwidth				200	kHz
	Slew Rate				0.2	$\text{V}/\mu\text{s}$
	Internal Sampling Frequency				3	kHz

$V^+ = 5\text{V}$, $V^- = -5\text{V}$, $V_{REF} = 0\text{V}$

V_{OS}	Gain Error (Note 2)	$A_V = 1$ ($R_L = 10\text{k}\Omega$)	●	-0.075	0	0.075 %
	Gain Error (Note 2)	$A_V = 2$ to 32 ($R_L = 10\text{k}\Omega$)	●	-0.5	0	0.5 %
	Gain Error (Note 2)	$A_V = 64$ to 1024 ($R_L = 10\text{k}\Omega$)	●	-0.6	-0.1	0.6 %
	Gain Error (Note 2)	$A_V = 2048$, 4096 ($R_L = 10\text{k}\Omega$)	●	-1	-0.2	1 %
	Gain Nonlinearity	$A_V = 1$	●		3	15 ppm
	Input Offset Voltage (Note 3)	$V_{CM} = 0\text{mV}$			5	± 20 μV
	Average Input Offset Drift (Note 3)	$T_A = -40^\circ\text{C}$ to 85°C $T_A = 85^\circ\text{C}$ to 125°C	● ●		± 50 ± 100	$\text{nV}/^\circ\text{C}$ $\text{nV}/^\circ\text{C}$
I_{OS}	Average Input Bias Current (Note 4)	$V_{CM} = 1\text{V}$	●	4	10	nA
	Average Input Offset Current (Note 4)	$V_{CM} = 1\text{V}$	●	1.5	3	nA

6915fb

ELECTRICAL CHARACTERISTICS The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at $T_A = 25^\circ\text{C}$.

CMRR	Common Mode Rejection Ratio	$A_V = 1024, V_{CM} = -5\text{V to } 5\text{V}, \text{ LTC6915C}$	●	105	123	dB
		$A_V = 1024, V_{CM} = -4.9\text{V to } 4.9\text{V, LTC6915I}$	●	105	123	
		$A_V = 1024, V_{CM} = -5\text{V to } 5\text{V, LTC6915I}$	●	100	123	
		$A_V = 1024, V_{CM} = -4.9\text{V to } 4.9\text{V, LTC6915H}$	●	100	123	
		$A_V = 1024, V_{CM} = -5\text{V to } 4.97\text{V, LTC6915H}$	●	90	123	
		$A_V = 1024, V_{CM} = -5\text{V to } 4.97\text{V, LTC6915H}$	●	90	123	
PSRR	Power Supply Rejection Ratio (Note 5)	$V_S = 2.7\text{V to } 11\text{V}$	●	110	116	dB
	Output Voltage Swing High	Sourcing 200 μA Sourcing 2mA	●	4.97	4.99	
	Output Voltage Swing Low	Sinking 200 μA Sinking 2mA	●	-4.98	-4.92	
	Supply Current, Parallel Mode	No Load, $V_{CM} = 0\text{mV}$	●	1.1	1.6	
	Supply Current, Serial Mode (Note 6)	No Load at OUT, Capacitive Load at $D_{OUT}(C_L) = 15\text{pF}$, Continuous CLK Frequency = 4MHz, CS = LOW, Gain Control Code = 0001	●	1.73	2.48	
	Supply Current, Shutdown	$V_{SHDN} = 4\text{V}$ (Hardware Shutdown) $V_{SHDN} = 1\text{V}$, Gain Control Code = 0000 (Software Shutdown)	●	160	25 240	μA
	SHDN Input High		●	4		
	SHDN Input Low		●	1	1	V

$V^+ = 5\text{V}, V^- = -5\text{V}, V_{REF} = 0\text{V}$

SHDN and HOLD_THRU Input Current (Note 2)		●	5	μA
Internal Op Amp Gain Bandwidth			200	kHz
Slew Rate			0.2	V/ μs
Internal Sampling Frequency			3	kHz

Digital I/O, All Digital I/O Voltage Referenced to DGND

V_{IH}	Digital Input High Voltage		●	2.0	V
V_{IL}	Digital Input Low Voltage		●	0.8	V
V_{OH}	Digital Output High Voltage	Sourcing 500 μA	●	$V^+ - 0.3$	V
V_{OL}	Digital Output Low Voltage	Sinking 500 μA	●	0.3	V
	Digital Input Leakage	$V^+ = 5\text{V}, V^- = -5\text{V}, V_{IN} = 0\text{V to } 5\text{V}$	●	± 2	μA

Timing, $V^+ = 2.7\text{V to } 4.5\text{V}, V^- = 0\text{V}$ (Note 7)

t_1	D_{IN} Valid to CLK Setup		●	60	ns
t_2	D_{IN} Valid to CLK Hold		●	0	ns
t_3	CLK Low		●	100	ns
t_4	CLK High		●	100	ns
t_5	\bar{CS}/LD Pulse Width		●	60	ns
t_6	LSB CLK to \bar{CS}/LD		●	60	ns
t_7	\bar{CS}/LD Low to CLK		●	30	ns
t_8	D_{OUT} Output Delay	$C_L = 15\text{pF}$	●	125	ns
t_9	CLK Low to \bar{CS}/LD Low		●	0	ns

LTC6915

ELECTRICAL CHARACTERISTICS

The ● denotes the specifications which apply over the full operating temperature range, otherwise specifications are at $T_A = 25^\circ\text{C}$.

Timing, $V^+ = 4.5\text{V}$ to 5.5V , $V^- = 0\text{V}$ (Note 7)

t_1	D_{IN} Valid to CLK Setup		●	30	ns
t_2	D_{IN} Valid to CLK Hold		●	0	ns
t_3	CLK Low		●	50	ns
t_4	CLK High		●	50	ns
t_5	\bar{CS}/LD Pulse Width		●	40	ns
t_6	LSB CLK to \bar{CS}/LD		●	40	ns
t_7	\bar{CS}/LD Low to CLK		●	20	ns
t_8	D_{OUT} Output Delay	$C_L = 15\text{pF}$	●	85	ns
t_9	CLK Low to \bar{CS}/LD Low		●	0	ns

Timing, Dual $\pm 4.5\text{V}$ to $\pm 5.5\text{V}$ Supplies (Note 7)

t_1	D_{IN} Valid to CLK Setup		●	30	ns
t_2	D_{IN} Valid to CLK Hold		●	0	ns
t_3	CLK High		●	50	ns
t_4	CLK Low		●	50	ns
t_5	\bar{CS}/LD Pulse Width		●	40	ns
t_6	LSB CLK to \bar{CS}/LD		●	40	ns
t_7	\bar{CS}/LD Low to CLK		●	20	ns
t_8	D_{OUT} Output Delay	$C_L = 15\text{pF}$	●	85	ns
t_9	CLK Low to \bar{CS}/LD Low		●	0	ns

Note 1: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. Exposure to any Absolute Maximum Rating condition for extended periods may affect device reliability and lifetime.

Note 2: These parameters are tested at $\pm 5\text{V}$ supply; at 3V and 5V supplies they are guaranteed by design.

Note 3: These parameters are guaranteed by design. Thermocouple effects preclude measurement of these voltage levels in high speed automatic test systems. V_{OS} is measured to a limit set by test equipment capability.

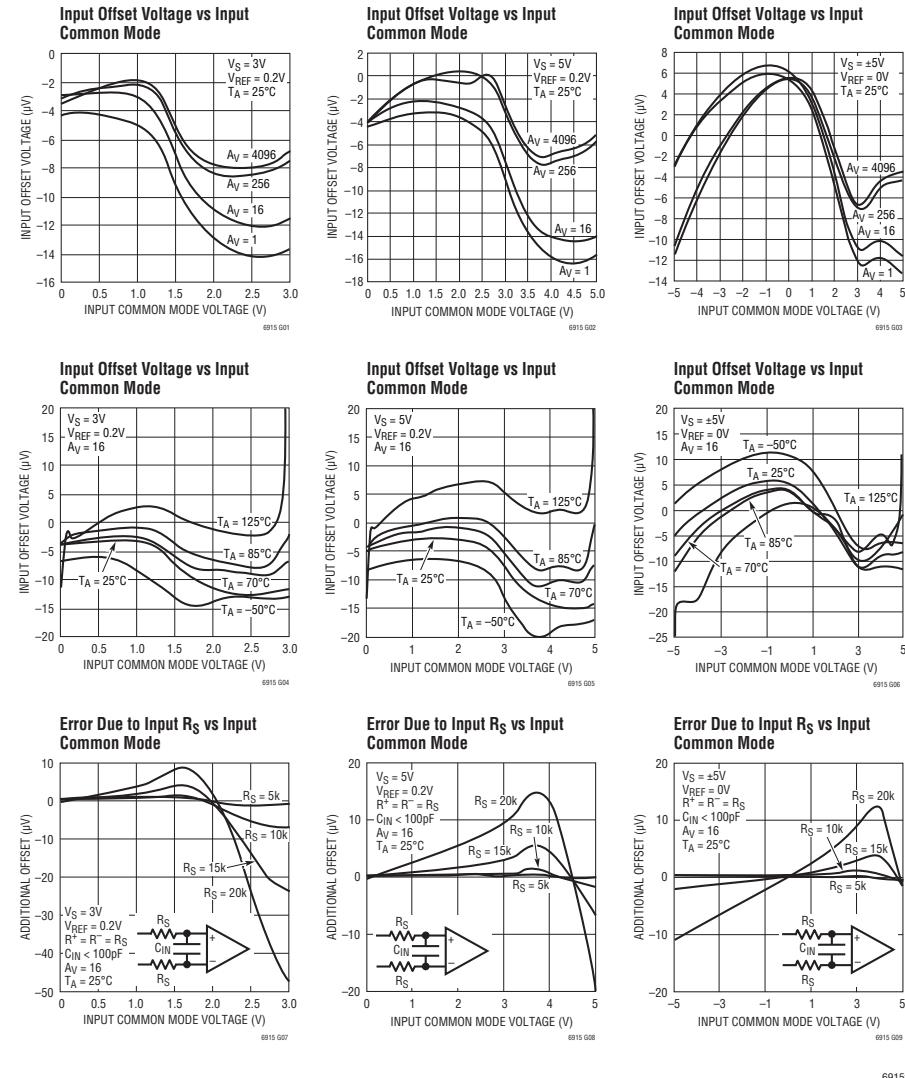
Note 4: If the total source resistance is less than 10k , no DC errors result from the input bias current or mismatch of the input bias currents or the mismatch of the resistances connected to IN^- and IN^+ .

Note 5: The PSRR measurement accuracy depends on the proximity of the power supply bypass capacitor to the device under test. Because of this, the PSRR is 100% tested to relaxed limits at final test. However, their values are guaranteed by design to meet the data sheet limits.

Note 6: Supply current is dependent on the clock frequency. A higher clock frequency results in higher supply current.

Note 7: Guaranteed by design, not subject to test.

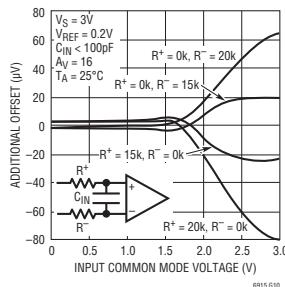
TYPICAL PERFORMANCE CHARACTERISTICS



LTC6915

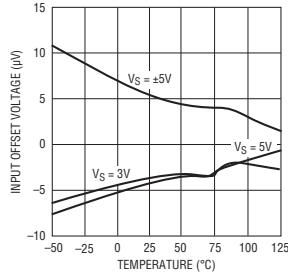
TYPICAL PERFORMANCE CHARACTERISTICS

Error Due to Input R_S Mismatch vs Input Common Mode



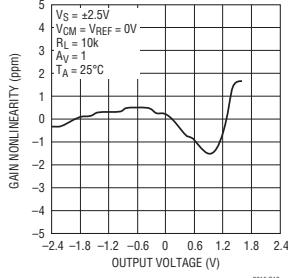
$V_S = 3\text{V}$, $V_{REF} = 0.2\text{V}$, $C_{IN} < 100\text{pF}$, $A_V = 16$, $T_A = 25^\circ\text{C}$, $R^+ = 0\text{k}, R^- = 15\text{k}$, $R^+ = 15\text{k}, R^- = 0\text{k}$, $R^+ = 20\text{k}, R^- = 0\text{k}$

Offset Voltage vs Temperature



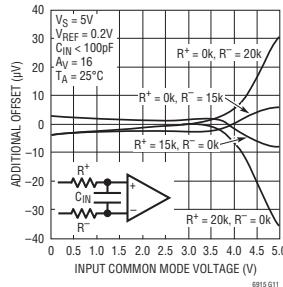
$V_S = \pm 5\text{V}$, $V_{REF} = 0\text{V}$, $C_{IN} < 100\text{pF}$, $A_V = 16$, $T_A = 25^\circ\text{C}$, $R^+ = 0\text{k}, R^- = 20\text{k}$, $R^+ = 15\text{k}, R^- = 15\text{k}$, $R^+ = 15\text{k}, R^- = 0\text{k}$, $R^+ = 20\text{k}, R^- = 0\text{k}$

Gain Nonlinearity at Gain = 1
(Gain Nonlinearity Decreases for
Gain > 1)



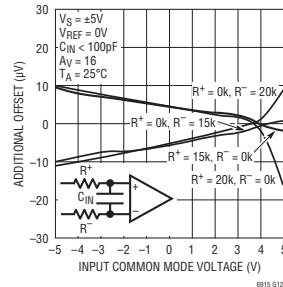
$V_S = \pm 2.5\text{V}$, $V_{CM} = V_{REF} = 0\text{V}$, $R_L = 10\text{k}$, $A_V = 1$, $T_A = 25^\circ\text{C}$

Error Due to Input R_S Mismatch vs Input Common Mode



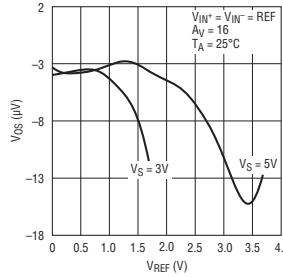
$V_S = \pm 5\text{V}$, $V_{REF} = 0.2\text{V}$, $C_{IN} < 100\text{pF}$, $A_V = 16$, $T_A = 25^\circ\text{C}$, $R^+ = 0\text{k}, R^- = 20\text{k}$, $R^+ = 15\text{k}, R^- = 15\text{k}$, $R^+ = 15\text{k}, R^- = 0\text{k}$, $R^+ = 20\text{k}, R^- = 0\text{k}$

Error Due to Input R_S Mismatch vs Input Common Mode



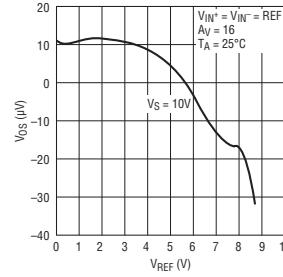
$V_S = \pm 5\text{V}$, $V_{REF} = 0\text{V}$, $C_{IN} < 100\text{pF}$, $A_V = 16$, $T_A = 25^\circ\text{C}$, $R^+ = 0\text{k}, R^- = 20\text{k}$, $R^+ = 15\text{k}, R^- = 15\text{k}$, $R^+ = 15\text{k}, R^- = 0\text{k}$, $R^+ = 20\text{k}, R^- = 0\text{k}$

V_{OS} vs REF (Pin 13)



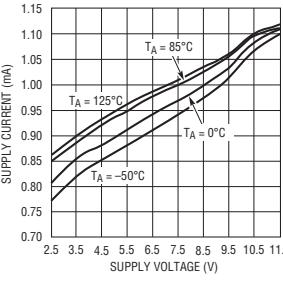
$V_{IN^+} = V_{IN^-} = \text{REF}$, $A_V = 16$, $T_A = 25^\circ\text{C}$

V_{OS} vs REF (Pin 13)



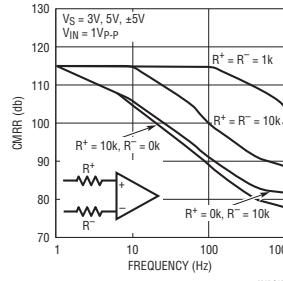
$V_{IN^+} = V_{IN^-} = \text{REF}$, $A_V = 16$, $T_A = 25^\circ\text{C}$

Supply Current vs Supply Voltage



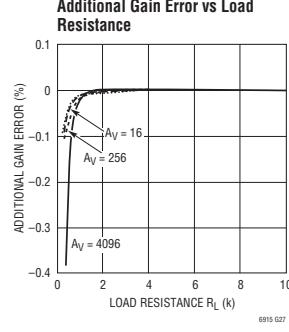
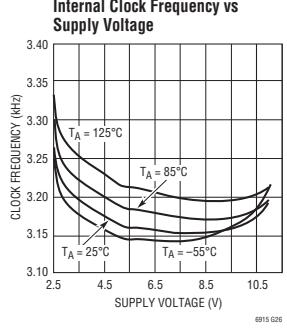
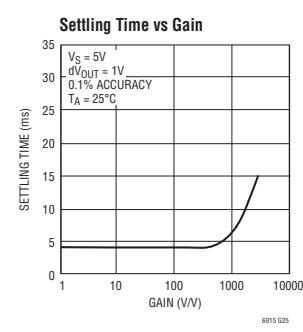
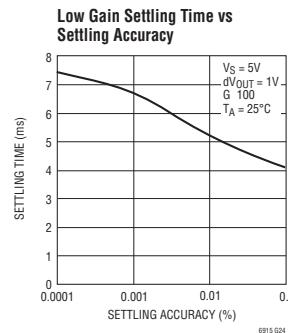
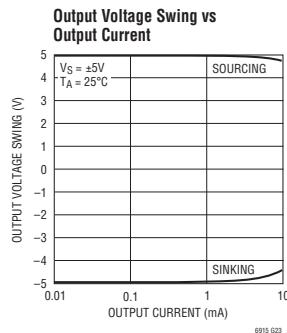
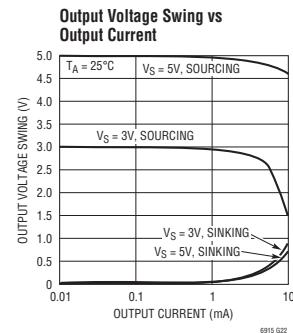
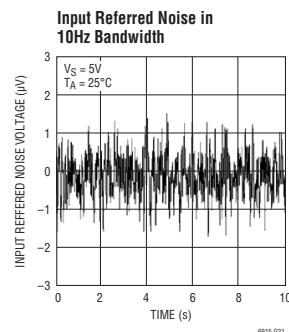
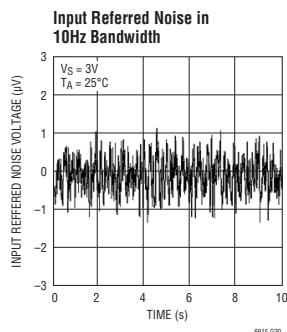
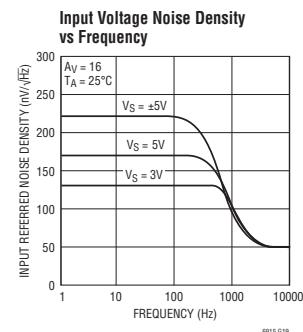
$V_S = 3\text{V}, \pm 5\text{V}$, $V_{IN} = 1\text{V}_{\text{P-P}}$, $R^+ = R^- = 1\text{k}$, $R^+ = R^- = 10\text{k}$, $R^+ = 10\text{k}, R^- = 0\text{k}$, $R^+ = 0\text{k}, R^- = 10\text{k}$

CMRR vs Frequency



$V_S = 3\text{V}, \pm 5\text{V}$, $V_{IN} = 1\text{V}_{\text{P-P}}$, $R^+ = R^- = 1\text{k}$, $R^+ = R^- = 10\text{k}$, $R^+ = 10\text{k}, R^- = 0\text{k}$, $R^+ = 0\text{k}, R^- = 10\text{k}$

TYPICAL PERFORMANCE CHARACTERISTICS



LTC6915

PIN FUNCTIONS (DFN/GN)

IN⁻ (Pin 1/Pin 2): Inverting Analog Input.

SHDN (Pin 1 GN Package Only): Shutdown Pin. The IC is shut down when SHDN is tied to V⁺. An internal current source pulls this pin to V⁻ when floating.

IN⁺ (Pin 2/Pin 3): Noninverting Analog Input.

V⁻ (Pin 3/Pin 4): Negative Supply.

CS(D0) (Pin 4/Pin 6): TTL Level Input. When in serial control mode, this pin is the chip select input (active low); in parallel control mode, this pin is the LSB of the parallel gain control code.

D_{IN}(D1) (Pin 5/Pin 7): TTL Level Input. When in serial control mode, this pin is the serial input data; in parallel mode, this pin is the second LSB of the parallel gain control code.

HOLD_THRU (Pin 5 GN Package Only): TTL Level Input for Parallel Control Mode. When HOLD_THRU is high, the parallel data is latched in an internal D-latch.

CLK(D2) (Pin 6/Pin 8): TTL Level Input. When in serial control mode, this pin is the clock of the serial interface; in parallel mode, this pin is the third LSB of the parallel gain control code.

D_{OUT}(D3) (Pin 7/Pin 9): TTL Level Input. When in serial control mode, this pin is the output of the serial data; in parallel mode, this pin is the MSB of the 4-bit parallel

gain control code. In parallel mode operation, if the data in to D_{OUT} (Pin 9) is from a voltage source greater than V⁺ (Pin 12), then connect a resistor between the voltage source and D_{OUT} to limit the current into Pin 9 to 5mA or less.

DGND (Pin 8/Pin 10): Digital Ground.

PARALLEL_SERIAL (Pin 9/Pin 11): Interface Selection Input. When tied to V⁺, the interface is in parallel mode, i.e., the PGA gain is defined by the parallel codes (D3 ~ D0), i.e., CS(D0), DATA(D1), CLK(D2), and D_{OUT}(D3). When PARALLEL_SERIAL pin is tied to V⁻, the PGA gain is set by the serial interface.

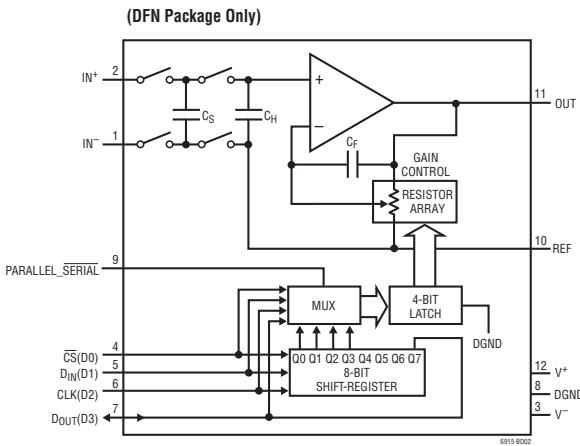
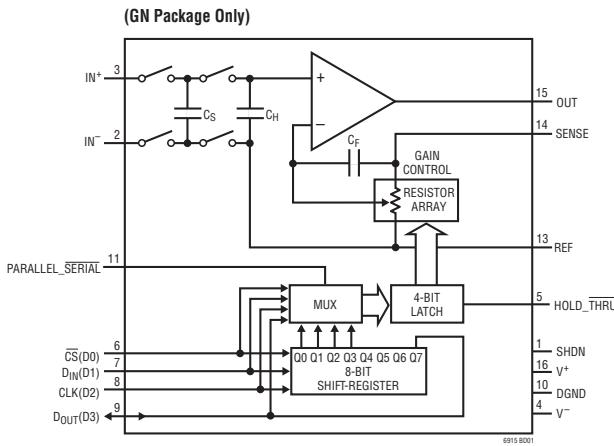
REF (Pin 10/Pin 13): Voltage Reference for PGA output.

OUT (Pin 11/Pin 15): Amplifier Output. The typical current sourcing/sinking of the OUT pin is 1mA. For minimum gain error, the load resistance should be 1k or greater (refer to the Output Voltage Swing vs Output Current and Gain Error vs Load Resistance in the Typical Performance Characteristics section).

V⁺ (Pin 12/Pin 16): Positive Supply.

SENSE (Pin 14 GN Package Only): Sense Pin. When the PGA drives a low resistance load and the interconnect resistance between the OUT pin and the load is not negligible, tying the SENSE pin as close as possible to the load can improve the gain accuracy.

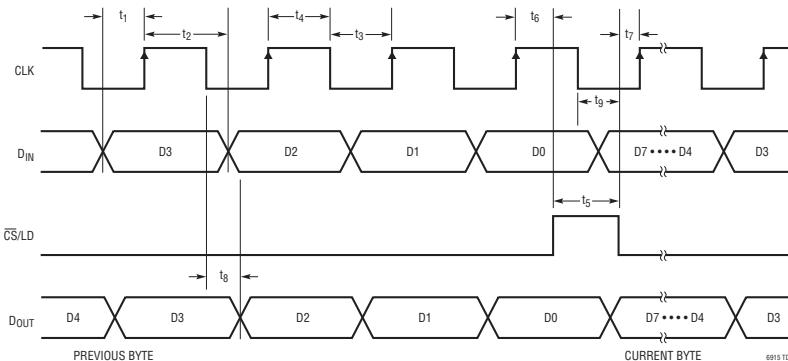
BLOCK DIAGRAMS



6915fb

LTC6915

TIMING DIAGRAM



OPERATION

Theory of Operation (Refer to Block Diagrams)

The LTC6915 uses an internal capacitor (C_S) to sample a differential input signal riding on a DC common mode voltage (the sampling rate is 3kHz and the input switch-on resistance is 1k to 2k, depending on the power supply voltage). This capacitor's charge is transferred to a second internal hold capacitor (C_H) translating the common mode voltage of the input differential signal to that of REF pin. The resulting signal is amplified by a zero-drift op amp in the noninverting configuration. Gain control within the amplifier occurs by switching resistors from a matched resistor array. The LTC6915 has 14 levels of gain, controlled by the parallel or serial interface. A feedback capacitor C_F helps to reduce the switching noise. Due to the input sampling, an LTC6915 may produce aliasing errors for input signals greater than 1.5kHz (one half the 3kHz sampling frequency). However, if the input signal is bandlimited to less than 1.5kHz then an LTC6915 is useful as instrumentation or as a differential to single-ended AC amplifier with programmable gain.

Parallel Interface

As shown in Figure 1, connecting PARALLEL_SERIAL to V⁺ allows the gain control code to be set through the parallel lines (D3, D2, D1, D0). When HOLD_THRU is

low (referenced to DGND) or floating, the parallel gain control bits (D3 ~ D0) directly control the PGA gain. When HOLD_THRU is high, the parallel gain control bits are read into and held by a 4-bit latch. Any change at D3 ~ D0 will not affect the PGA gain when HOLD_THRU is high. Note that the DFN12 package does not have the HOLD_THRU pin. Instead, it is tied to DGND internally. The D_{OUT}(D3) pin is bidirectional (output in serial mode, input in parallel mode). In parallel mode, the voltage at D_{OUT}(D3) cannot exceed V⁺; otherwise, large currents can be injected to V⁺ through the parasitic diode (see Figure 2). Connecting a 10k resistor at the D_{OUT}(D3) pin if parallel mode is selected (see Figure 1) is recommended for current limiting.

Serial Interface

Connecting PARALLEL_SERIAL to V⁻ allows the gain control code to be set through the serial interface. When CS is low, the serial data on D_{IN} is shifted into an 8-bit shift-register on the rising edge of the clock, with the MSB transferred first (see Figure 3). Serial data on D_{OUT} is shifted out on the clock's falling edge. A high CS will load the 4 LSBs of the shift-register into a 4-bit D-latch, which are the gain control bits. The clock is disabled internally when CS is pulled high. Note: CLK must be low before CS is pulled low to avoid an extra internal clock pulse.

6915fb

OPERATION

D_{OUT} is always active in serial mode (never tri-stated). This simplifies the daisy chaining of the multiple devices. D_{OUT} cannot be “wire-or” to other SPI outputs. In addition, D_{OUT} does not return to zero at the end of transmission, i.e. when \overline{CS} is pulled high.

A LTC6915 may be daisy-chained with other LTC6915s or other devices having serial interfaces by connecting

the D_{OUT} to the D_{IN} of the next chip while CLK and \overline{CS} remain common to all chips in the daisy chain. The serial data is clocked to all the chips then the \overline{CS} signal is pulled high to update all of them simultaneously. Figure 4 shows an example of two LTC6915s in a daisy chained SPI configuration.

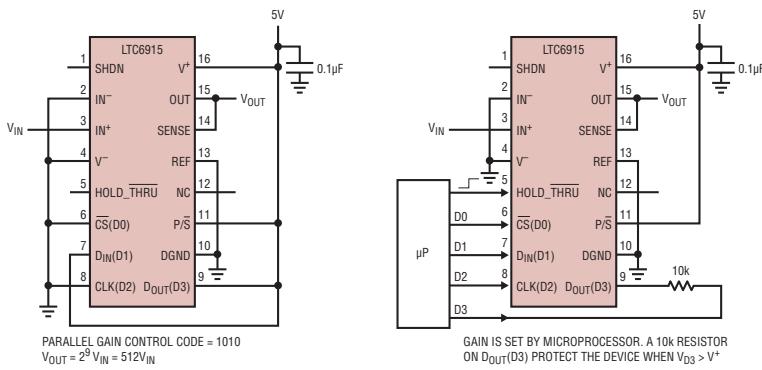


Figure 1. PGA in the Parallel Control Mode

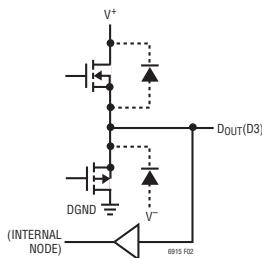


Figure 2. Bidirectional Nature of $D_{OUT}/D3$ Pin

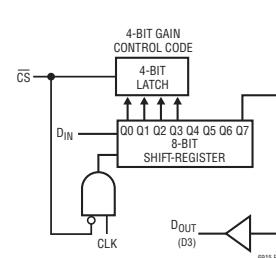


Figure 3. Diagram of Serial Interface (MSB First Out)

LTC6915

OPERATION

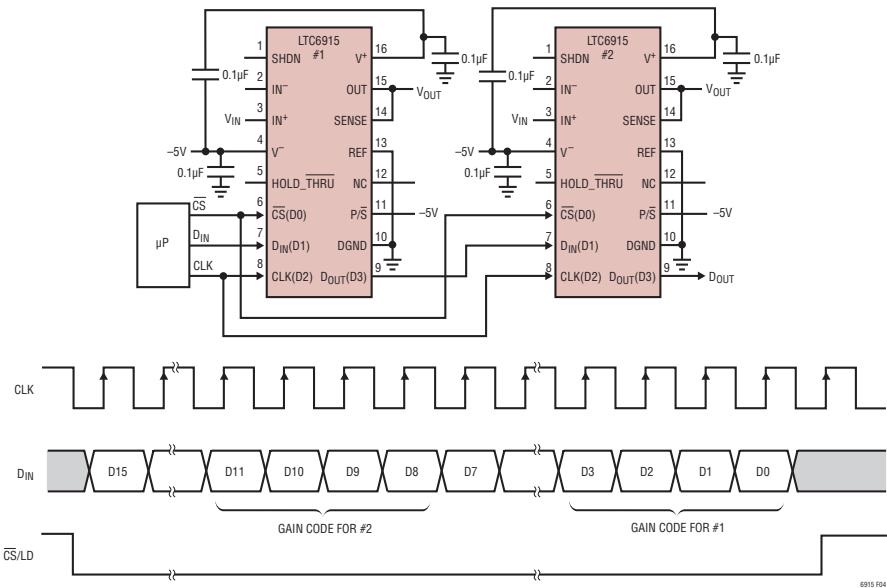


Figure 4.2 PGAs in a Daisy Chain

The amplifier's gain is set as follows:

D3, D2, D1, D0	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101-1111
Gain	0	1	2	4	8	16	32	64	128	256	512	1024	2048	4096

Input Voltage Range

The input common mode voltage range of the LTC6915 is rail-to-rail. However, the following equation limits the size of the differential input voltage:

$$V^- \leq (V_{IN^+} - V_{IN^-}) + V_{REF} \leq V^+ - 1.3$$

Where V_{IN^+} and V_{IN^-} are the voltage of the differential input pins, V^+ and V^- are the positive and negative supply voltages respectively and V_{REF} is the voltage of REF pin. In addition, V_{IN^+} and V_{IN^-} must not exceed the power supply voltages, i.e.,

$$V^- < V_{IN^+} < V^+ \text{ and } V^- < V_{IN^-} < V^+$$

±5 Volt Operation

When using the LTC6915 with supplies over 5.5V, care must be taken to limit the maximum difference between any of the input pins (IN^+ or IN^-) and the REF pin to 5.5V, i.e.,

$$|V_{IN^+} - V_{REF}| < 5.5 \text{ and } |V_{IN^-} - V_{REF}| < 5.5$$

If not, the device will be damaged. For example, if rail-to-rail input operation is desired when the supplies are at ±5V, the REF pin should be 0, ±0.5V. As a second example, if the V^+ pin is 10V, and the V^- and REF pins are at 0, the inputs should not exceed 5.5V.

LTC6915

TYPICAL APPLICATION

Multiplexing Two LTC6915's

Send a gain code of 0000 to one IC to set its output to a high impedance state and send a gain code other than 0000 to the second IC to set it for normal amplification. If both devices are ON, the 200Ω resistors protect the outputs. The sense pin connection maintains gain accuracy for loads 1k or greater.

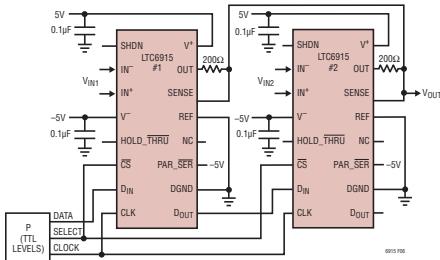
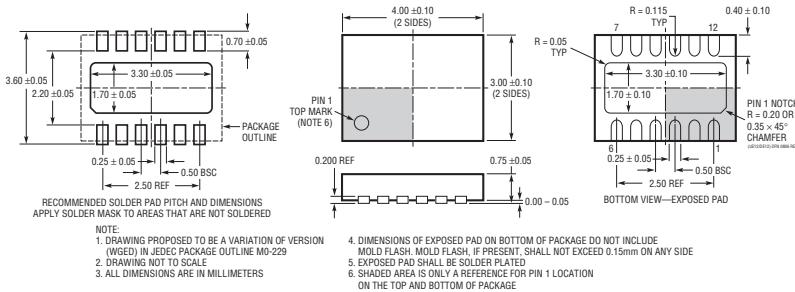


Figure 6. A 2:1 Multiplexing Two LTC6915's with Daisy Chained Gain Control

PACKAGE DESCRIPTION

DE/UE Package 12-Lead Plastic DFN (4mm × 3mm) (Reference LTC DWG # 05-08-1695 Rev D)

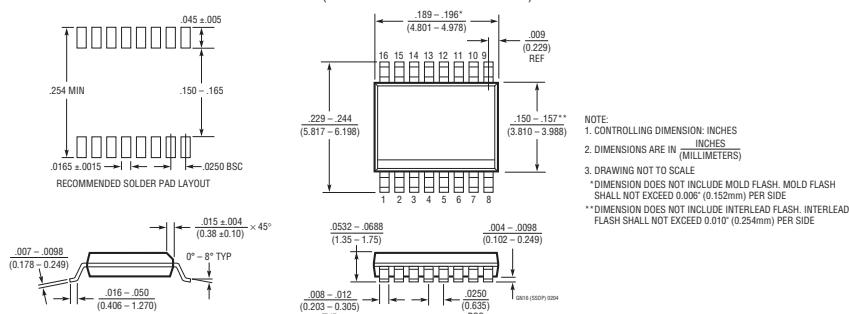


NOTE:

1. DRAWING PROPOSED TO BE A VARIATION OF VERSION (WG6) IN JEDEC PACKAGE OUTLINE MO-229
2. DRAWING NOT TO SCALE
3. ALL DIMENSIONS ARE IN MILLIMETERS

4. DIMENSIONS OF EXPOSED PAD ON BOTTOM OF PACKAGE DO NOT INCLUDE MOLD FLASH. MOLD FLASH, IF PRESENT, SHALL NOT EXCEED 0.15mm ON ANY SIDE
5. EXPOSED PAD SHALL BE SOLDER PLATED
6. SHADeD AREA IS ONLY A REFERENCE FOR PIN 1 LOCATION ON THE TOP AND BOTTOM OF PACKAGE

GN Package 16-Lead Plastic SSOP (Narrow .150 inch) (Reference LTC DWG # 05-08-1641)



NOTE:
1. CONTROLLING DIMENSION: INCHES
2. DIMENSIONS ARE IN (MILLIMETERS)
3. DRAWING NOT TO SCALE
*DIMENSION DOES NOT INCLUDE MOLD FLASH. MOLD FLASH SHALL NOT EXCEED 0.006" (0.152mm) PER SIDE
**DIMENSION DOES NOT INCLUDE INTERLEAD FLASH. INTERLEAD FLASH SHALL NOT EXCEED 0.010" (0.254mm) PER SIDE

6915fb

REVISION HISTORY (Revision history begins at Rev B)

REV	DATE	DESCRIPTION	PAGE NUMBER
B	6/11	Revised units for PSRR in Electrical Characteristics	5



Information furnished by Linear Technology Corporation is believed to be accurate and reliable. However, no responsibility is assumed for its use. Linear Technology Corporation makes no representation that the interconnection of its circuits as described herein will not infringe on existing patent rights.

6915fb

17

LTC6915

TYPICAL APPLICATION

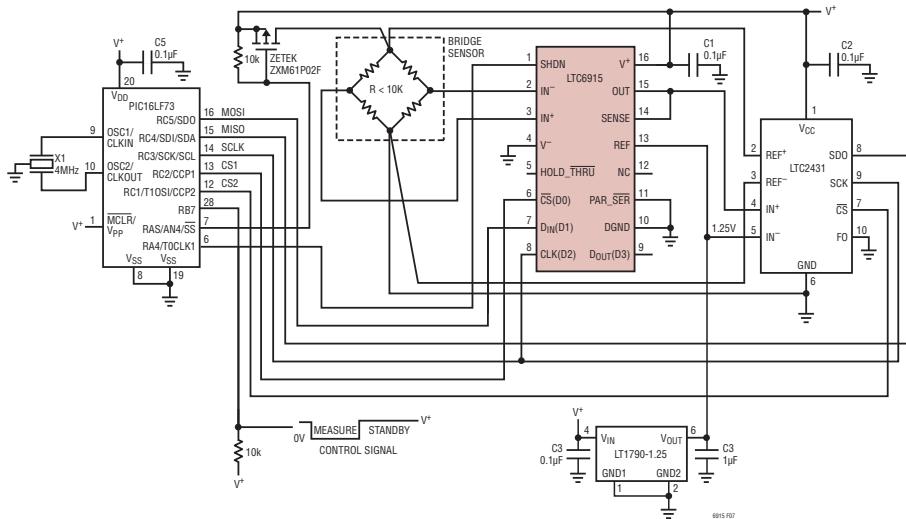


Figure 7. Bridge Amplifier with Programmable Gain and Analog to Digital Conversion. (Standby Current Less than 100µA)

RELATED PARTS

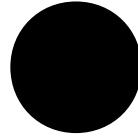
PART NUMBER	DESCRIPTION	COMMENTS
LTC1043	Dual Precision Instrumentation Switched-Capacitor Building Block	Rail-to-Rail Input, 120dB CMRR
LTC1100	Precision Zero-Drift Instrumentation Amplifier	Fixed Gains of 10 or 100, 10µV Offset, 50pA Input Bias Current
LTC1101	Precision, Micropower, Single Supply Instrumentation Amplifier	Fixed Gain of 10 or 100, I _S < 105µA
LTC1167	Single Resistor Gain Programmable, Precision Instrumentation Amplifier	Single Gains Set Resistor, G = 1 to 10,000 Low Noise: 7.5nV/√Hz
LTC1168	Low Power Single Resistor Gain Programmable, Precision Instrumentation Amplifiers	I _S = 530µA
LTC1789-1	Single Supply, Rail-to-Rail Output, Micropower Instrumentation Amplifier	I _S = 80µA Max
LTC2050	Zero-Drift Operational Amplifier	SOT-23 Package
LTC2051	Dual Zero-Drift Operational Amplifier	MS8 Package
LTC2052	Quad Zero-Drift Operational Amplifier	GN16 Package
LTC2053	Rail-to-Rail Input and Output, Zero-Drift Instrumentation Amplifier with Resistor-Programmable Gain	MS8 Package, 10µV Max V _{OS} , 50nV/°C Max Drift
LTC6800	Rail-to-Rail Input and Output, Instrumentation Amplifier with Resistor-Programmable Gain	MS8 Package, 100µV Max V _{OS} , 250nV/°C Max Drift

6915fb

Document Information

Project reference

317947, FP7-ICT-2011-8



Context

Deliverable 2.1, Workpackage 2

Document title

201309_Arduino_Sands_Deliverable_2-1_TWI-protocol

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

Davey Taylor

Contact information

d.taylor@arduino.cc

Document created

1. Aug. 2013

Last modification

1 Sep. 2013

Table of contents

Document Information.....	2
Table of contents.....	3
TWI protocol	4
IDENT/ALARM enhancement.....	4
Command summary.....	5
Generic commands.....	5
Relay module commands.....	5
Consumption module commands.....	5
Mosfet module commands.....	5
Input module commands.....	5
Generic commands.....	6
ID_START	6
ID_END	6
ID_PASS	6
ID_TEMP	6
ID_READ.....	6
ID_WRITE.....	7
DEVINFO	7
Relay commands.....	8
STATE_SET	8
STATE_GET	8
PWM_SET.....	8
PWM_GET	8
CONFIG_SET	8
CONFIG_GET	8
Mosfet commands.....	9
READ	9
STATE_GET.....	9
PWM_SET	9
STATE_GET.....	9
CONFIG_SET	9
CONFIG_GET.....	10
Consumption commands.....	11
READ	11
CONFIG_SET	11
CONFIG_GET	11
Input commands.....	12
READ_ANALOG	12
READ_DIGITAL.....	12
SET_BIAS	12
GET_BIAS	12
SET_GAIN.....	12
GET_GAIN	12
CONFIG_SET	12
CONFIG_GET.....	13
Shield commands.....	14
BEGIN	14
TX	14
RTS	14

TWI protocol

The SandS modules can communicate through a modified version of a standard implementation of the two wire interface (TWI), also called I2C in the Sands documentation.

This protocol enhances the standard I2C specification using two extra pins, labeled IDENT (identification) and ALARM that can be used for adding extra functionality to the modules. In this case the two new features are:

- a self identification protocol, through the IDENT pin, that will allow assigning standard I2C addresses to the different modules while they are all connected in series to the SandS motherboard
- an alarm pin that will be used for the different boards (motherboard and modules) to inform about an abnormal operation

All of the boards implement the generic set of commands of the TWI protocol internally as part of the firmware. On top of that, each one of the modules implements a series of board-specific commands.

If you were about to make modifications to the basic firmware to any of the modules, you should start from the standard SandS firmware provided by Arduino SA for whatever module you are using, if you want to keep the functionality explained in this document.

IDENT/ALARM enhancement

The IDENT pin behaviour is described in further detail under the GENERIC COMMANDS section.

The ALARM pin is a multi-drop signal which is kept high by means of a pull-up resistor on the host board. This signal will be pulled low by a connected module if the programmed alarm criteria for that module is met.

Since this signal is shared between all modules, as well as the host, it enables immediate detection and response to critical problems in the system without the need for constantly polling all modules.

Modules can be programmed to immediately respond to an alarm on their own, without requiring host interaction - enabling critical functions to be shut down quickly even if the host board has been damaged or is otherwise unable to recover the system.

Command summary

Generic commands

The list of generic communication commands defined in the firmware for all boards is described in the following table.

ID_START	Enter identification mode
ID_END	Exit identification mode
ID_PASS	Pass token to next device
ID_TEMP	Set temporary TWI-address
ID_READ	Read the programmed address
ID_WRITE	Program a new address into device
DEVINFO	Read device information

Table 1: list of generic TWI commands

Relay module commands

The commands for the relay module are as follows.

STATE_SET	Set output state
STATE_GET	Read output state
PWM_SET	Set PWM duty cycle
PWM_GET	Get PWM duty cycle
CONFIG_SET	Set device config.
CONFIG_GET	Get device config.

Table 2: Relay board specific commands

Consumption module commands

The commands for the consumption module are as follows.

READ	Read consumption
CONFIG_SET	Set device config.
CONFIG_GET	Get device config.

Table 3: Consumption board specific commands

Mosfet module commands

The commands for the mosfet module are as follows.

READ	Read current
STATE_SET	Set ouput state
STATE_GET	Read output state
PWM_SET	Set PWM duty cycle
PWM_GET	Get PWM duty cycle
CONFIG_SET	Set device config.
CONFIG_GET	Get device config.

Table 4: Mosfet board specific commands

Input module commands

The commands for the input module are as follows.

READ_ANALOG	Read analog input val.
READ_DIGITAL	Read digital input val.
SET_BIAS	Set bias voltage
GET_BIAS	Read current bias volt.
SET_GAIN	Set voltage gain
GET_GAIN	Read voltage gain
CONFIG_SET	Set device config.
CONFIG_GET	Get device config.

Table 5: Input board specific commands

Generic commands

ID_START

Use this command to enter identification mode, thus to change the TWI address of a module, read its address, and other related functions.

This command should be sent to the general-call address **0x00**.

This is known as putting the devices in **ID MODE**, and allows the host to enumerate and identify all connected devices, as well as program them with unique TWI-addresses for further communication.

A device will accept **ID MODE** commands only when the upstream (towards host) **IDENT** pin is driven high by the previous device (or, for the first device in the chain, the host) and the downstream (away from host) is not yet driven high by the device (see the **ID_PASS** command).

This condition is known as owning the token, and issuing the **ID_PASS** command is known as passing the token.

By this scheme, only one unit can own the token at any given time, which also implies that only one device is able to accept **ID MODE** commands at any given time.

All other (non **ID MODE**) commands will be accepted either if the device is in **ID MODE** and owns the token, or the device is in normal operation and is addressed either using the general call address or the devices programmed address.

Note: it is not possible to read data from a device using the general call address.

Command sequence (1 byte):

0x80

ID_END

Use this command to exit identification mode and return the module to standard operation.

This command should be sent to the general-call address **0x00**.

When devices receive this command they will exit **ID MODE** and return to normal operation.

If the device has a temporary address it will revert to its programmed address.

Command sequence (1 byte):

0x81

ID_PASS

This command will pass the identification token to the next module in the chain.

This is an **ID MODE** command and will only be accepted by the device owning the token.

This command should be sent to the general-call address **0x00**.

When a device receives this command it will pass the token to the next device in the chain.

The process takes roughly 100 milliseconds, after which the downstream (away from host) **IDENT** pin will be driven high by the device.

Command sequence (1 byte):

0x82

ID_TEMP

It will set a temporary TWI-address to the module.

This is an **ID MODE** command and will only be accepted by the device owning the token.

This command should be sent to the general-call address **0x00**.

When a device receives this command, it will temporarily change it's TWI-address to the issued address.

This is useful when the host needs to read data from a device that has no TWI-address, since it is not possible to read data from a device that does not have a TWI-address.

Immediately after issuing this command it is possible for the host to read data from the device, using the issued address.

The originally programmed address can still be read using the **ID_READ** command, which is useful for finding the address of unknown devices in the chain.

The recommended address for temporary addressing is **0x7F**.

The device will revert to its programmed address when it passes the token to the next device, or when it exits **ID MODE** (see the **ID_END** command).

Command sequence (2 bytes):

0x85 [Temporary address:8]

ID_READ

Used to read the module's current programmed address.

When a device receives this command, it will respond with its **[Programmed address]**.

If a device has been issued a temporary address (see the **ID_TEMP** command) it will still respond with its **[Programmed address]** and not the temporary address.

Command sequence (1 byte):

0x83

Response (1 byte):

[Programmed address:8]

[Device type:8] [Major:8] [Minor:8] [Revision:8]

ID_WRITE

This command will program a [New address] into the device.

The device will respond with a success code:

0x01	Success
0x80	Failure

Table 6: ID Write response values

If programming is successful the device will immediately switch to the [New address], and will no longer respond to commands at its previous address, or the temporary address if one has been issued (see the ID_TEMP command).

However, if programming is unsuccessful the device will not switch to the [New address] and it is up to the host to implement a scheme for detecting the success/failure condition.

Command sequence (2 bytes):

0x85 [New address:8]

Response (1 byte):

[Success code:8]

DEVINFO

When issuing this command, the device will respond with its device information.

All response bytes are in ASCII, including the version number ranging from 0.0.1 to 9.9.9

[Device type] is defined in the following table:

R	Relay board
M	MOSFET board
I	Input board
C	Consumption board
S	Comm. Board / shield

Table 7: Possible answers to DEVINFO

[Major], [Minor] and [Revision] describe the version of the board, and are ASCII characters ranging from '0' to '9'.

Command sequence (1 bytes):

0x00

Response (4 bytes):

Relay commands

STATE_SET

This command sets the output [State] of the device to one of three values:

0x00	Relay is off
0x01	Relay is on
0x80	Relay controlled by PWM

Table 8: Possible SET states, relay module

Command sequence (2 bytes):

0x01 [State:8]

STATE_GET

The device will respond with the current [State] value:

0x00	Relay is off
0x01	Relay is on
0x80	Relay is controlled by PWM and is currently off
0x81	Relay is controlled by PWM and is currently on

Table 9: Possible GET states, relay module

Command sequence (1 byte):

0x02

Response (1 byte):

[State:8]

PWM_SET

This command sets the [Duty] cycle for PWM mode to a value between **0x00** and **0xFF** representing the range 0..100%.

If the output state (see the **STATE_SET** command) is not PWM the output will not be affected, but the value will take effect if the device state is later set to PWM.

Command sequence (2 bytes):

0x03 [Duty:8]

PWM_GET

The device will respond with the currently set [Duty]

cycle value.

Command sequence (1 byte):

0x04

Response (1 byte):

[Duty:8]

CONFIG_SET

This command will program a new configuration for the device.

The configuration is non-volatile and will be remembered if the device loses power.

The configuration will only be programmed if all values are in the accepted range.

If the configuration is successfully programmed, it will also take effect immediately.

For valid [Power on state]s, see the **STATE_SET** command.

The [PWM period] (seconds) must be in the range **0x01** to **0xF0**.

[Comm Timeout] (milliseconds) specifies the amount of time allowed between successfully received messages before raising an alarm – 0 = disabled.

[Response Mode] specifies how to respond to alarms (local or from other devices). The device will return to normal operation as soon as the alarm is de-asserted.

0x00	No response, continue operation as normal
0x01	Force output off
0x80	Force output on

Table 10: Configuration response mode values

Command sequence (6 bytes):

0x20 [Power on state:8] [PWM period:8]
[Comm Timeout:16] [Response Mode:8]

CONFIG_GET

The device will respond with the currently programmed configuration.

See the **CONFIG_SET** command for a description of the response parameters.

Command sequence (1 byte):

0x21

Response (5 bytes):

[Power on state:8] [PWM period:8]
[Comm Timeout:16] [Response Mode:8]

Mosfet commands

READ

The device will respond with the [**Current**] (milliamperes) running through the MOSFET in, as an unsigned 16-bit integer.

Note that the precision of the readout is roughly 20mA, but the readout may contain an offset error of up to 100mA and a scaling error of up to 3%, meaning that a readout of 1000mA may actually correspond to any value between 873mA and 1133mA. This can be corrected for by calibration, if higher accuracy is required.

Command sequence (1 byte):

0x01

Response (2 bytes):

[**Current:16**]

STATE_SET

This command sets the output [**State**] of the device to one of three values:

0x00	Mosfet is off
0x01	Mosfet is on
0x80	Mosfet controlled by PWM

Table 11: Possible SET states, Mosfet module

Command sequence (2 bytes):

0x03 [Mode:8]

STATE_GET

The device will respond with the current [**State**] value:

0x00	Mosfet is off
0x01	Mosfet is on
0x80	Mosfet controlled by PWM

Table 12: Possible GET states, Mosfet module

Command sequence (1 byte):

0x04

Response (1 byte):

[**State:8**]

PWM_SET

This command sets the [**Duty**] cycle for PWM mode to a value between **0x00** and **0xFF** representing the range 0..100%.

If the output state (see the **STATE_SET** command) is not PWM the output will not be affected, but the value will take effect if the device state is later set to PWM.

Command sequence (2 bytes):

0x05 [Duty:8]

STATE_GET

The device will respond with the currently set duty cycle value.

Command sequence (1 byte):

0x06

Response (1 byte):

[**Duty:8**]

CONFIG_SET

This command will program a new configuration for the device.

The configuration is non-volatile and will be remembered if the device loses power.

The configuration will only be programmed if all values are in the accepted range.

If the configuration is successfully programmed, it will also take effect immediately.

For valid [**Power on state**], see the **STATE_SET** command.

The [**PWM frequency**] is specified in decahertz, in the range **0x01** to **0xFA**, representing 10Hz to 2.5kHz.

[**Comm Timeout**] (milliseconds) specifies the amount of time allowed between successfully received messages before raising an alarm – 0 = disabled.

[**Alarm Mode**] specifies the level alarm mode of operation:

0x00	Disabled
0x01	Alarm when [Current] below [Alarm Level]
0x02	Alarm when [Current] above/equal to [Alarm Level]

Table 13: ALARM modes, Mosfet module

[Alarm Level] specifies the threshold (corresponds to **[Current]**, see **READ** command) level for alarm. This has no effect if **[Alarm Mode]** is disabled, or if Mode is digital.

[Alarm Delay] (milliseconds) specifies the amount of time that the above conditions must be met before asserting the alarm output.

[Response Mode] specifies how to respond to alarms (local or from other devices). The device will return to normal operation as soon as the alarm is de-asserted. Possible modes are:

0x00	No response, continue operation as normal
0x80	Force output off
0x81	Force output on

Table 14: Response modes, Mosfet module

Command sequence (11 bytes):

0x20 [Power on state:8] [PWM frequency:8]
 [Comm Timeout:16] [Alarm Mode:8]
 [Alarm Level:16] [Alarm Delay:16]
 [Response Mode:8]

CONFIG_GET

The device will respond with the currently programmed configuration.

See the **CONFIG_SET** command for a description of the response parameters.

Command sequence (1 byte):

0x21

Response (10 bytes):

[Power on state:8] [PWM frequency:8]
 [Comm Timeout:16] [Alarm Mode:8]
 [Alarm Level:16] [Alarm Delay:16]
 [Response Mode:8]

Consumption commands

READ

The device will respond with the current power **[Consumption]** (milliamperes), as an unsigned 16-bit integer.

Note: the precision of the readout is highly dependant on the configuration.

Command sequence (1 byte):

0x01

Response (2 bytes):

[Consumption:16]

CONFIG_SET

This command will program a new configuration for the device.

The configuration is non-volatile and will be remembered if the device loses power.

The configuration will only be programmed if all values are in the accepted range.

[Range] is specified in Amperes, given that the split-core sensor outputs 0.333V at 100% of it's rated current, and must be in the range **0x01** to **0x32**, representing 1A to 50A.

[Comm Timeout] (milliseconds) specifies the amount of time allowed between successfully received messages before raising an alarm – 0 = disabled.

[Alarm Mode] specifies the level alarm mode of operation:

0x00	Disabled
0x01	Alarm when [Consumption] below [Alarm Level]
0x02	Alarm when [Consumption] above/equal to [Alarm Level]

Table 15: ALARM modes, Consumption module

[Alarm Level] specifies the threshold (corresponds to **[Consumption]**, see **READ** command) level for alarm. This has no effect if **[Alarm Mode]** is disabled, or if Mode is set to digital.

[Alarm Delay] (milliseconds) specifies the amount of time that the above conditions must be met before asserting the alarm output.

Command sequence (9 bytes):

0x20 [Range:8] [Comm Timeout:16]
[Alarm Mode:8] **[Alarm Level:16]** **[Alarm Delay:16]**

CONFIG_GET

The device will respond with the currently programmed configuration.

See the **CONFIG_SET** command for a description of the response parameters.

Command sequence (1 byte):

0x21

Response (8 bytes):

[Range:8] **[Comm Timeout:16]** **[Alarm Mode:8]**
[Alarm Level:16] **[Alarm Delay:16]**

Input commands

READ_ANALOG

The device will respond with the current [Readout], as an unsigned 16-bit integer in the range **0x0000** to **0xFFFF**.

Command sequence (1 byte):

0x01

Response (2 bytes):

[Readout:16]

READ_DIGITAL

The device will respond with the current [Readout]:

0x80	Input is LOW
0x81	Input is HIGH

Table 16: Readout values, Input module

Command sequence (1 byte):

0x02

Response (1 byte):

[Readout:8]

SET_BIAS

This will set the bias voltage of the input device.

[Bias] is an unsigned 16-bit integer in the range **0x0000** to **0x03FF**.

This will not update the non-volatile configuration memory of the device and if the device loses power, it will revert to the programmed bias (see the **CONFIG_SET** command).

Command sequence (3 bytes):

0x03 [Bias:16]

GET_BIAS

The device will respond with the current bias voltage setting.

See the **BIAST_SET** command for a description of the response parameters.

Command sequence (1 byte):

0x04

Response (2 bytes):

[Bias:16]

SET_GAIN

This will set the voltage gain of the input device.

[Gain] is in the range **0x01** to **0xD** and describes the requested gain according to:

$$RealGain = 2^{[Gain]-1}$$

Formula 1: [Gain] vs RealGain

This will not update the non-volatile configuration memory of the device and if the device loses power, it will revert to the programmed gain (see the **CONFIG_SET** command).

Command sequence (2 bytes):

0x05 [Gain:8]

GET_GAIN

The device will respond with the current voltage gain setting.

See the **GAIN_SET** command for a description of the response parameters.

Command sequence (1 byte):

0x06

Response (1 byte):

[Gain:8]

CONFIG_SET

This command will program a new configuration for the device.

The configuration is non-volatile and will be remembered if the device loses power.

The configuration will only be programmed if all values are in the accepted range.

[Mode] is specified as follows:

0x00	Analog, voltage 0 ... 3.3V
0x01	Analog, voltage 0 ... 12 V
0x02	Analog, voltage 0 ... 24 V
0x03	Analog, resistive sensor
0x10	Digital, DC 0 ... 3.3V
0x11	Digital, DC 0 ... 12 V
0x12	Digital, DC 0 ... 24 V

0x20	Digital, AC 0 ... 3.3V
0x21	Digital, AC 0 ... 12 V
0x22	Digital, AC 0 ... 24 V

Table 17: Possible modes, input board

See the **BIAS_SET** command for a description of the **[Bias]** parameter.

See the **GAIN_SET** command for a description of the **[Gain]** parameter.

[Comm Timeout] (milliseconds) specifies the amount of time allowed between successfully received messages before raising an alarm – 0 = disabled.

[Alarm Mode] specifies the level alarm mode of operation:

0x00	Disabled
0x01	Alarm when [Readout] below [Alarm Level] (analog) or is 0 (digital)
0x02	Alarm when [Readout] above/equal to [Alarm Level] (analog) or is 1 (digital)

Table 18: ALARM modes, Input module

[Alarm Level] specifies the threshold (corresponds to **[Readout]**, see **READ_ANALOG** command) level for alarm. This has no effect if **[Alarm Mode]** is disabled, or if Mode is set tp digital.

[Alarm Delay] (milliseconds) specifies the amount of time that the above conditions must be met before asserting the alarm output.

Command sequence (12 bytes):

0x20 [Mode:8] [Bias:16]
 [Gain:8] [Comm Timeout:16]
 [Alarm Mode:8] [Alarm Level:16] [Alarm Delay:16]

CONFIG_GET

The device will respond with the currently programmed configuration.

See the **CONFIG_SET** command for a description of the response parameters.

Command sequence (1 byte):

0x21

Response (11 bytes):

[Mode:8] [Bias:16] [Gain:8]
 [Comm Timeout:16] [Alarm Mode:8]
 [Alarm Level:16] [Alarm Delay:16]

Shield commands

Following any device specific command, this device will always respond with received data. This means a device specific command must be sent after any generic commands, or the response to the generic command will be repeated. Sending a TX command with no data will suffice.

Command sequence (2 byte):

0x03 [RTS:8]

[CTS&Bytecount] contains the current state of the serial port CTS pin in its most significant bit, and the number of bytes that follow in the remaining 7 bits.

Response sequence (1+n bytes):

[CTS&Bytecount8] [Data:8*Bytecount]

BEGIN

The device will configure the serial port with the specified settings.

[Baudrate] is a 16-bit word containing the baudrate in bits per second.

[Standard] must be one of the following:

0x00	RS232
0x02	RS485 Full Duplex
0x03	RS485 Half Duplex

Table 19: Communication Standards

If [SlewLimit] is non-zero, slew rate limiting will be enabled, otherwise it will be disabled.

Command sequence (6 bytes):

0x01 [Baudrate:16] [Format:8] [Standard:8]
[SlewLimit:8]

TX

The device will transmit the specified data on the serial port.

Command sequence (1+n bytes):

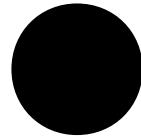
0x02 [Data:8*n]

RTS

This command is only valid for RS232 communication, and will configure the RTS pin on the serial port

If [RTS] is non-zero, the RTS pin will be asserted, otherwise it will be deasserted.

Document Information



Project reference

317947, FP7-ICT-2011-8

Context

Deliverable 2.1, workpackage 2

Document title

201309_Arduino_Sands_Deliverable_2-1_UI

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

1. Sep. 2013

Table of contents

Document Information.....	1
Table of contents.....	2
Summary.....	3
Purpose within SandS.....	3
SandS UI module.....	4
AVR core.....	4
Power the board.....	4
Hardware.....	5
Board dimensions.....	6
Mounting considerations.....	6
Schematic.....	6
Board file.....	6
Pin layout.....	6
Firmware.....	6
Licenses.....	7
Hardware.....	7
Firmware.....	7
Appendix 1 - Schematics.....	8

Summary

This document describes the main characteristics of the SandS UI (User Interface) module, an open source hardware - Arduino compatible board. It comes with a 5V serial port to connect to the SandS motherboard directly, but also a USB port via a micro USB connector for debugging and firmware updates. This board displays the information sent by the motherboard, but also can handle the use of images via the on-board SD card.

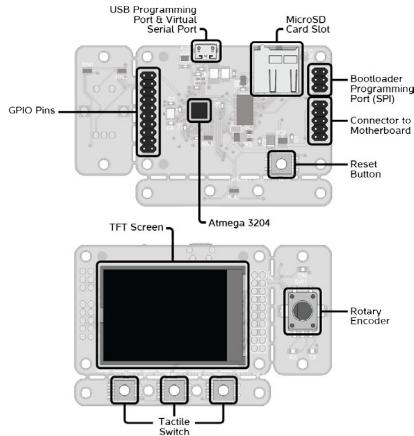


Image 1: block description, UI module

The use scenario for this board is to offer a user interface to manipulate the SandS appliance program directly and do e.g. configuration operations in an easy way. Therefore it comes with a rotary encoder and three tactile switches to be used to control the interface.

The board runs from the power provided by the Serial port. It can be reprogrammed from the Arduino IDE using a microUSB cable and the SandS special library. It comes with a pre-loaded firmware.

Even if the board has been specifically designed for the SandS project, the pins exposed directly from the processor (the ATmega32U4) can be addressed as Arduino digital pins.

The TFT display used is a 7 inch display that is part of the standard Arduino catalogue. It comes with a controller that allows addressing its 160x120 pixels with a color depth of 16 bits. The standard Arduino TFT library allows controlling this device as well as loading BMP images directly from the microSD card it carries on its back.

Purpose within SandS

The purpose of this module is to offer a user interface for showing information about the appliance, the connectivity to the network, or the use of a certain recipe on the appliance.

SandS UI module

The SandS UI module is a microcontroller board operating as a I2C peripheral to the SandS motherboard. It is made of two blocks:

- a microcontroller (the ATmega32U4): running at 16MHz it listens to the serial port and executes the different operations on the module
- a series of connectors where to plug the different modules

It comes with a microUSB port, one 5V RS-232 port, and the SPI programming port to burn the processor's bootloader. See Image 1 for a block diagram of the board. It also exposes a series of GPIO pins on the processor on a two row connector header.

AVR core

The AVR core chosen for the SandS relay module is the ATmega32U4, which main characteristics are described in Table 1:

Processor	ATmega32U4
Flash Memory	32 KB of which 4 KB used by bootloader
SRAM	2,5 KB
EEPROM	1 KB
Operating Voltage	5 V
Hardware Serial ports	1
Hardware SPI ports	1
Hardware I2C ports	1
Native USB ports	1

Table 1: main characteristics of the AVR core

This processor can be programmed as an Arduino Leonardo board directly from the standard Arduino IDE, there is more information how to program the board on the *Getting Started* guide to the module.

The ATmega32U4 has an internal USB peripheral, therefore, it can communicate directly with a computer via a standard USB port.

Power the board

Input Voltage (recommended)	5 V through the Serial or USB connector
Current (min)	50 mA

Table 2: powering the board

It is possible to power the board from the USB port or, alternatively, from the serial port connector used to communicate to the SandS motherboard.

Hardware

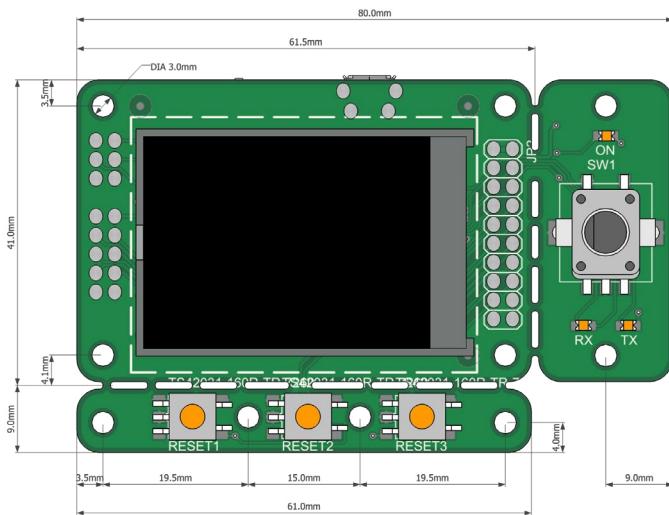


Image 2: Top view, Sands UI module v0002



Image 3: Isometric view of the SandS UI module v0002

Board dimensions

The board has been designed to easily accommodate the parts in a 2-layer design in the size of 50x80mm. It comes with 3mm holes at different locations, making it easy to mount it using standard plastic spacers and M3 screws.

Check Image 2 for further details about measurements and location of the holes on the board.

You will notice that the board has been designed to be separated into pieces if needed.

Image 3 gives you an idea of the proportions of the different components on the board.

Mounting considerations

It is recommended to mount the board in a plastic box and, if possible, inside the same box as the motherboard.

The module will be connected serially to the motherboard. This means that the board will use standard board-to-cable connectors, but the cables needed will not have to exit the box.

Schematic

The board's schematic shows all the components on it. You can check all the schematics at Appendix 1, at the end of this document.

The schematics come as a single document showing:

- SD card connector
- TFT display connector
- USB connector
- three switches
- one rotary encoder
- a series of GPIO pins
- Microcontroller: is the ATmega32U4, with a 16Mhz crystal
- the SPI connector for reprogramming the processor's bootloader
- the serial port to communicate with the motherboard
- the serial port connection for reprogramming the device and debugging its firmware

Board file

The board file is a 2-layered design with the components separated strategically to avoid potential interferences.

Images 2 and 3 give a pretty good idea of this layout. Both schematic and board file for this design come

bundled in the same zipped file as this document and can be modified using the Eagle Cad software.

Pin layout

Image 4 shows the pin layout for the board as well as the description of each one of the pins separately.

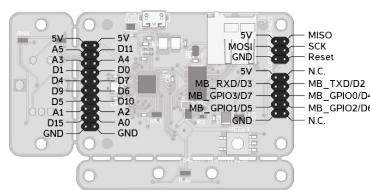


Image 4: pin layout, UI module

Firmware

The SandS UI board comes with a preprogrammed firmware that will get the processor to receive commands through the serial port. The normal operation mode doesn't need of reprogramming the device. All the specific commands for this device are described standard Arduino documentation for the Arduino TFT display .

The firmware is open source and comes installed in the board. It is possible to reprogram it directly from the Arduino IDE. It is also possible to use other tools like command line, or AVR studio to change the firmware in the processor.

1 <http://arduino.cc/en/Reference/TFTLibrary>

Licenses

Hardware

The design of the boards falls under the CERN Open Hardware License 1.2, for more information refer to the following link:

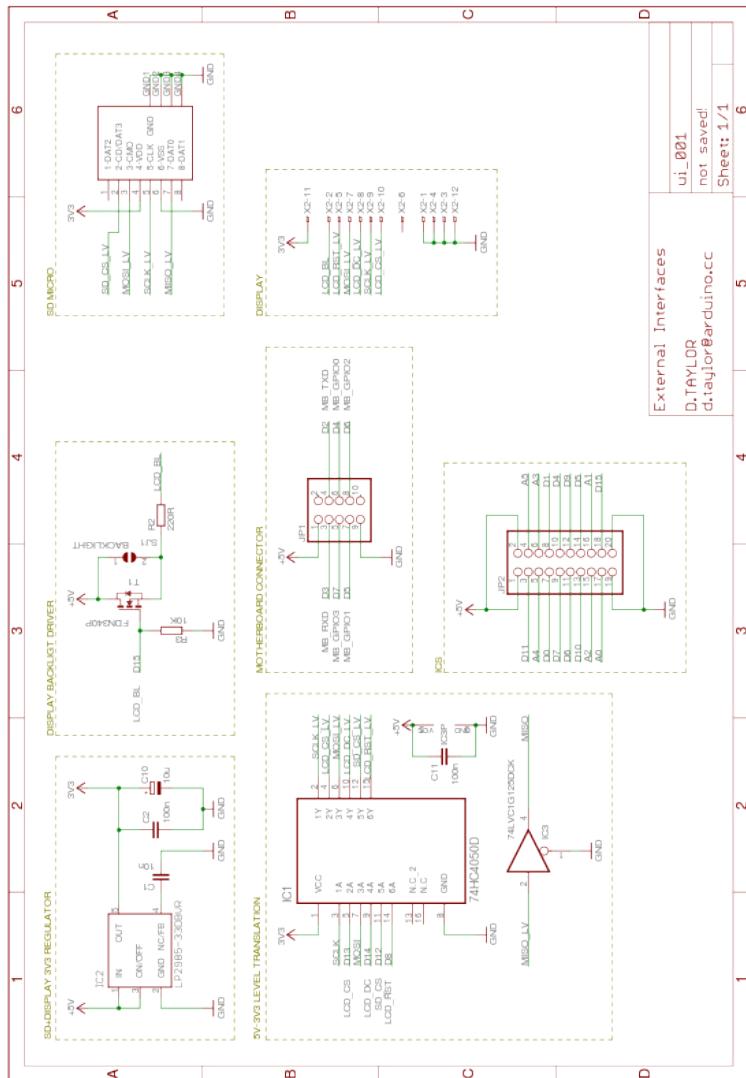
http://ohwr.org/attachments/2388/cern_ohl_v_1_2.txt

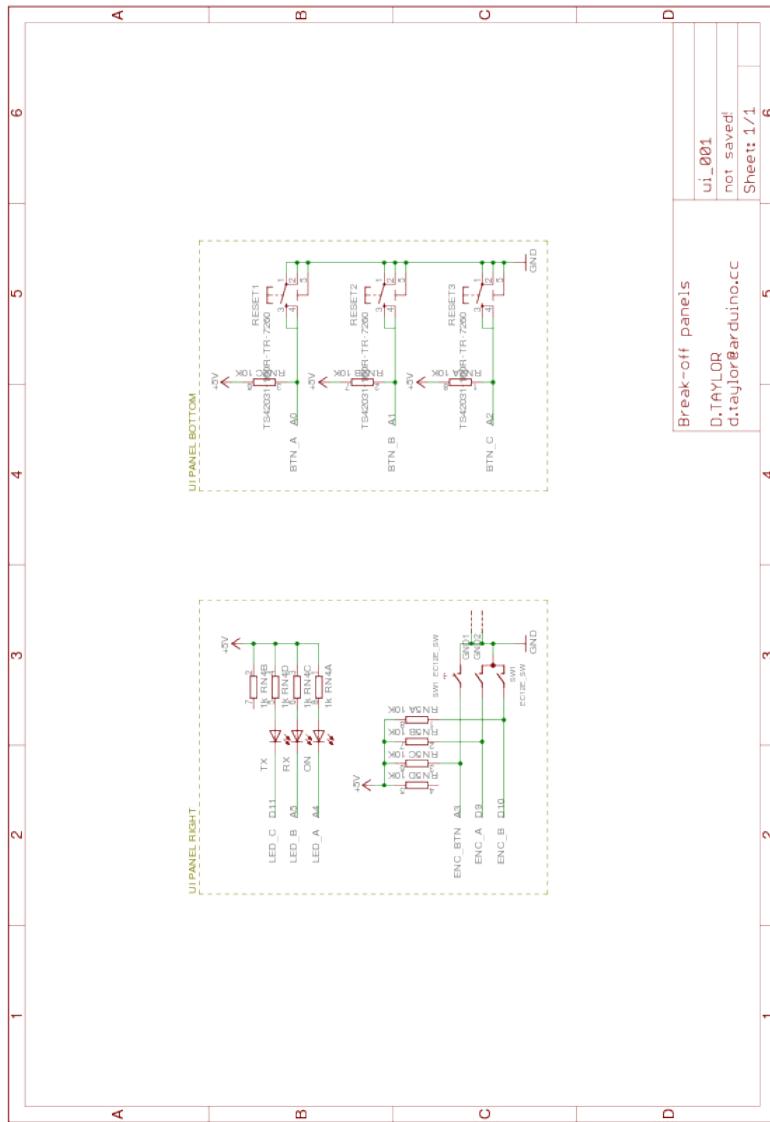
Firmware

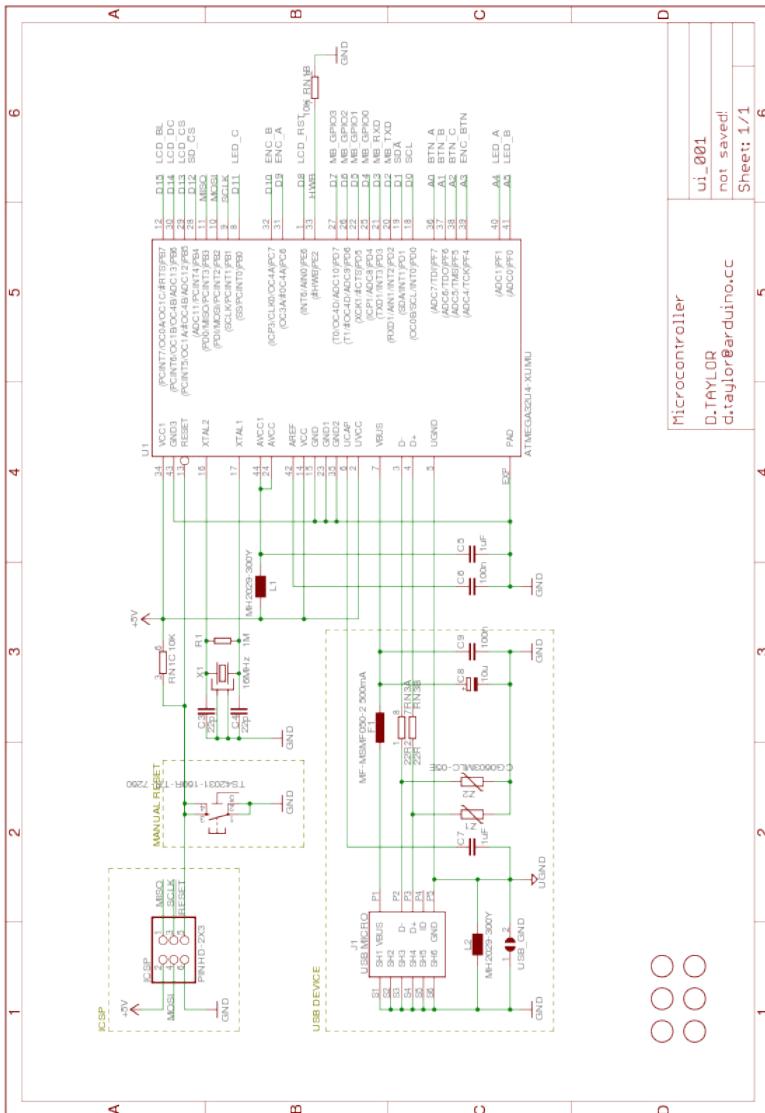
The firmware for the ATmega32U4 processor used in the SandS UI board is based on the Arduino core and therefore it is licensed under LGPL, however all the source is available for anybody to use. To read more about this license, refer to:

<https://www.gnu.org/licenses/lgpl.html>

Appendix 1 - Schematics









APPENDIX C, PUBLICATION 2

Appendix C, Publication 2

D. Cuartielles. *Delivery number D2.2: Report on Thinking Appliance Manual* (Malmö: Social&Smart, 2014a). [SandS].



SandS: Social & Smart

Social housekeeping through intercommunicating appliances
and shared recipes merged in a pervasive web-services
infrastructure

Delivery number D2.2. Report on thinking appliance manual

Project	
Project acronym:	SOCIAL&SMART
Project Full title:	Social housekeeping through intercommunicating appliances and shared recipes merged in a pervasive web-services infrastructure
Grant agreement no.:	317947
Document	
Deliverable number	D2.2
Deliverable title:	Thinking appliance manual
Workpackage:	WP2
Due date of deliverable:	31/08/2013
Lead beneficiary:	Arduino
Editors:	D. Cuartielles
Contributing beneficiaries:	ARD, GORE, LBL
Reviewer:	B. Apolloni
Status:	Final version
Version and date:	V04, 09/03/2014
Changes:	Editorial changes

Project co-funded under the SEVENTH FRAMEWORK PROGRAMME THEME 3 ICT -
INFORMATION AND COMMUNICATIONS TECHNOLOGIES

Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission Services)	PP
RE	Restricted to a group specified by the Consortium (including the Commission Services)	
CO	CO Confidential, only for members of the Consortium (including the Commission Services)	

Project co-ordinator: Prof. Bruno Apolloni
Università degli studi di Milano
Tel: +39 02 50316284
E-mail: apolloni@di.unimi.it
Project website address: www.sands-project.eu



Publishable summary

This document summarizes the way how the SandS hardware ecosystem can be deployed inside an appliance and programmed to connect to the internet and, by extension, to the SandS infrastructure.

This deliverable builds on top of D2.1 that was dedicated to describing the SandS hardware. In this case, we focus in the ways the different elements can be configured and reprogrammed. It also describes the tools created specifically for the project in order to reprogram the appliances.

Together with this deliverable, we provide a series of documents that look in more detail into the different parts of the system. The following documents are referred to throughout the deliverable:

- *D2-2_Getting-Starter-Guide_SandS_motherboard.pdf*: explains how to configure the drivers and how to upload Arduino code to the SandS motherboard
- *D2-2_Getting-Started-Guide_SandS_I2C_configuration_tool.pdf*: explains how to connect a series of modules to a SandS motherboard and configure them from a command line interface
- *D2-2_Getting-Started-Guide_SandS_cross_compiler.pdf*: explains how to create an SDK to compile code that can be uploaded to the Linux part of the SandS motherboard
- *D2-2_Built_In_Oven_Remote_Extension_Protocol.pdf*: specifies the communication over serial port towards Gorenje's HomeChef oven. This serves as an example of how to get the SandS motherboard to talk to devices that offer a serial port connection



CONTENTS

Publishable summary.....	2
CONTENTS.....	3
1. Introduction.....	4
2. The SandS workflow.....	5
3. Programming the SandS motherboard.....	8
4. Configuring the modules.....	9
5. Creating code for the Linux core of the motherboard.....	11
6. Example: Connecting to the HomeChef frontpanel.....	11
7. Example: Connecting to the SandS bread-making machine.....	12
8. Conclusions.....	12



1. Introduction

The goal of this deliverable is the creation of a series of manuals that will help the SandS user getting started in the use of the SandS hardware in order to equip any kind of appliance. This report summarizes the main aspects of all of the documents generated as a result of working with tasks T2.1 to T2.4.



2. The SandS workflow

The goal of this deliverable is providing anyone having an interest in connecting a certain appliance to the SandS infrastructure, with the understanding on how to get started using the open source tools designed specifically for this purpose.

While D2.1 was dealing with the hardware design, D2.2 is more focused in the different software tools needed to configure, program and use the SandS hardware ecosystem. The question is: which are the steps to be taken in order to approach the idea of equipping or “hacking” an appliance?

The steps are simple and can be described in the following list:

1. select the device you want to equip
2. identify the physical actuators and sensors you need to provide and decide which modules from the SandS selection in Figure 1 will be the best ones in each case
3. connect the modules to the hardware and then each module in a chain that starts on the SandS motherboard
4. plug the motherboard in a PC using a USB cable and don't forget to power the SandS motherboard
5. run the SandS Configurator, a CLI tool to configure the addresses and other characteristics of the modules in use
6. create your own state machine to control the appliance as you wish/need
7. connect your motherboard to the WiFi

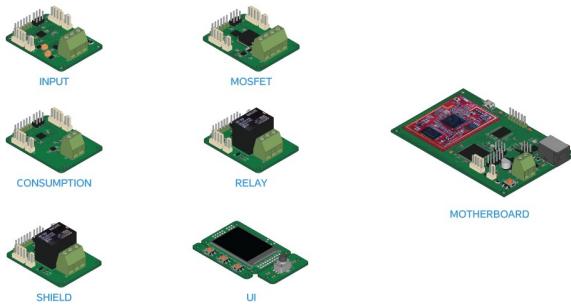


Figure 1: SandS hardware ecosystem, all of the boards designed for the project are programmable from the Arduino IDE



To simplify the comprehension of these series of actions, we have created the diagram presented on Figure 2, where it is possible to see which is the best way to attack the problem of intervening a pre-fabricated device. However, it could also be applied to the creation of an entirely new appliance using the SandS motherboard as the core.

Explaining each one of the different parts of this workflow is the main aim of D2.2. In this case the beneficiaries ARD, LBL and GOR have collaborated in searching for solutions to connect devices to the net, meanwhile developing flexible-enough hardware to accommodate as many approaches as possible.



THE SANDS WORKFLOW

HOW TO MAKE CONNECTED APPLIANCES

1 CHOOSE THE APPLIANCE



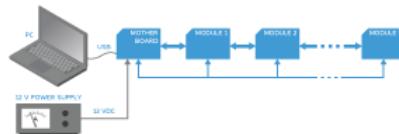
2 IDENTIFY WHICH ARE THE PHYSICAL CONTROLS TO INTERVENE AND CHOOSE THE SANDS MODULES YOU NEED TO CONNECT THEM.



3 CONNECT ALL MODULES TO THE MOTHERBOARD



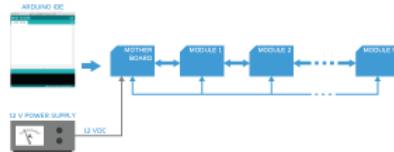
4 PLUG THE MOTHERBOARD TO A PC USING A USB CABLE AND THEN POWER THE BOARD



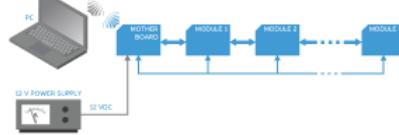
5 RUN THE CONFIGURATION TOOL



6 UPLOAD YOUR STATE MACHINE TO CONTROL YOUR APPLIANCE



7 CONNECT TO YOUR MOTHERBOARD OVER WI-FI AND CONFIGURE THE CONNECTION



8 START USING YOUR APPLIANCE ON THE SANDS INFRASTRUCTURE

Figure 2: the SandS workflow, or how to make connected appliances



3. Programming the SandS motherboard

The SandS motherboard is an Arduino compatible device. This means that it can be programmed using the standard Arduino IDE and all of the functions in the Arduino programming language. Figure 3 recalls the motherboard main functionalities.

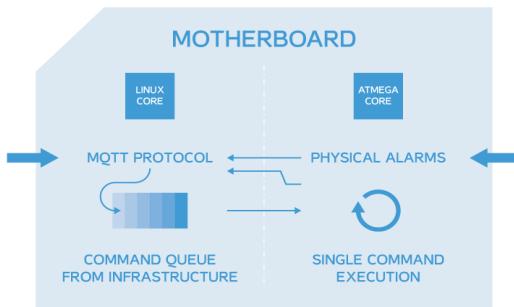


Figure 3: functional view of the SandS motherboard

There was the need to produce the libraries to ease the work to be done to handle the communication when the board is connected either to modules or to appliances via a serial port, like the HomeChef from Gorenje.

Preparing a computer to program the board's AVR core – which is responsible for the low level drivers that control motors, relays, and sensors – is a two steps process:

1. you need to download Arduino's IDE from the Arduino website: <http://arduino.cc>
2. you need the libraries provided as part of this deliverable to be uncompressed inside a folder called “libraries”. The latter should be placed in the location where your Arduino IDE stores your programs

The two libraries created for the project are:

1. SandS: a library that contains a whole series of commands to run with the *configurator* and handle communication at low level between the modules and the motherboard
2. HomeChef: a library that comprises commands to accommodate the serial port protocol defined by Gorenje to control their oven

There is more information about the way the motherboard can be programmed in the document: *D2-2_Getting-Starter-Guide_SandS_motherboard.pdf*.



4. Configuring the modules

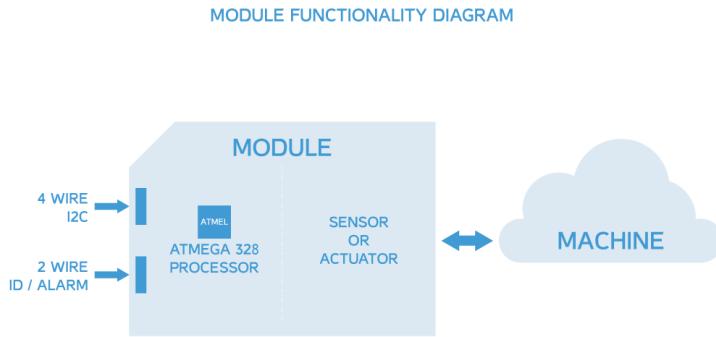


Figure 4: functionality diagram of a generic SandS module

The SandS modules come pre-programmed with their respective firmware. It is possible to modify their firmware not only via a USB-serial converter, but also from the SandS motherboard using the SandS Two Wire Protocol, which is, in a way, an implementation of I2C. Internally, the modules include a small AVR processor that will handle the communication as well as actions' execution on the sensors and actuators. While Figure 4 illustrate this aspect form a functional perspective, Figure 5 gives an example of how the configurator works.



```
C:\Windows\system32\cmd.exe - cfg COM7
C:\sands>cfg COM7
Opening connection... OK!
Initializing... OK!
> target 03
Targetting INPUT v1.0.0, ID is 03
> config
To read, write or show configuration:
    config read/write/show
To change configuration:
    config [parameter]
> config show
mode Mode Analog, resistive sensor
bias Bias 0
gain Gain 1
atime Comm timeout 0ms
amode Alarm mode alarm disabled
alevel Alarm level 0mA
adelay Alarm delay 500ms
> config gain
Gain <gain, 1-13 - see manual>: 2
> config write
Writing 11 bytes... done
>
```

Figure 5: configurator in action, in this case chesign and modifying the basic configuration of an input module

In order to simplify the work of configuring the I2C addresses and other internal parameters of the modules, we have sketched in Figure 6 a command line tool that allows configuring all modules and dynamically assign new addresses to them.

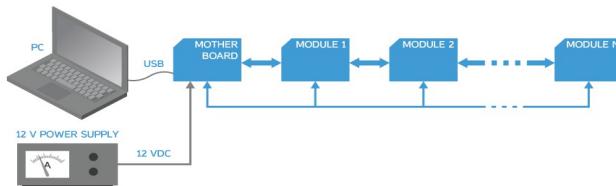


Figure 6: setup needed to run the configurator tool

There is more information about these aspects in the file: *D2-2_Getting-Started-Guide_SandS_I2C_configuration_tool.pdf*.



5. Creating code for the Linux core of the motherboard

Yet another complex task in this part of the project was to create the documentation to be able of setting up a cross-compiling tool. This is needed in order to compile code in the Linux core of the board for the different partners (mainly LBL and GOR). The goal is to create the communication libraries needed to establish the communication towards the SandS infrastructure.

The series of steps to follow in order to create the cross-compilation toolchain, aren't too complex. But this activity is very time consuming.

The document: *D2-2_Getting-Started-Guide_SandS_cross_compiler.pdf* explains how to prepare a Linux 64 computer to act as a compilation tool for your own linux apps to be running on the SandS motherboard.

6. Example: Connecting to the HomeChef frontpanel

Gorenje provided Arduino with one of their smart oven front panel. The SandS motherboard connects to the smart Gorenje's appliances via a 3v3 serial port. Gorenje also created an API for Arduino to access to their ovens in a simple way.

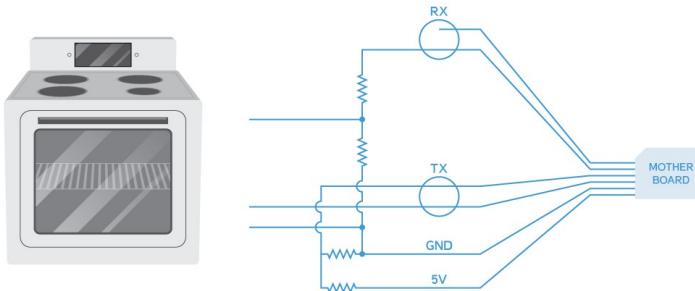


Figure 7: connecting to the HomeChef using a resistor bridge

The connection between the SandS motherboard and HomeChef by Gorenje is done using a simple resistor bridge which giving us great results (as shown on the data of the stress test done for deliverable D2.3). See Figure 7.

Arduino's role was to design the HomeChef library -mentioned earlier- to support connecting to the oven. That API is registered in the document *D2-2_Built_In_Oven_Remote_Extension_Protocol.pdf*.



7. Example: Connecting to the SandS bread-making machine

The SandS bread-making machine is a hacked bread-making machine incorporating the SandS motherboard, or compatible board. There are multiple ways of intervening an existing appliance, but in this case, we chose to directly hack: 1. the machine's motor, 2. the heating element, and 3. the temperature control.

Figure 8 shows one of the possible ways to intervene this machine.

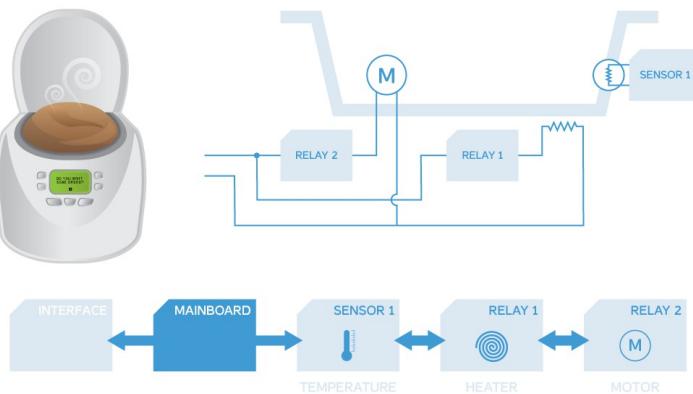
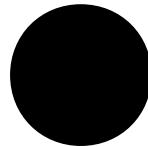


Figure 8: bread making machine hacked using the SandS motherboard

8. Conclusions

The manuals produced as a part of this deliverable are basically showing that it is possible to create a generic piece of hardware, in this case the SandS motherboard, that can become useful for filling a whole series of different cases that we may confront when trying to connect an appliance to the internet.

Document Information



Project reference

317947, FP7-ICT-2011-8

Context

Deliverable 2.2, workpackage 2

Document title

D2-2_Built_In_Oven_Remote_Extension_Protocol

Company information

*Arduino SA
Via D'Alberti, 1
6830 Chiasso
Switzerland*

Document version number

v0001

Author

David Cuartielles

Contact information

dcuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

9. Mar. 2014

D2.2 Gorenje built-in oven remote extension protocol (GOR)

Table of Contents

Document Information.....	1
D2.2 Gorenje built-in oven remote extension protocol (GOR).....	2
1 Introduction.....	2
2 Communication.....	2
2.1 Pinout.....	2
3 Frame structure.....	4
4 Functional profile.....	4
4.1 Status function.....	4
4.2 Internal light switching.....	5
4.3 Operation control.....	6
4.4 Recipe upload.....	6

1 Introduction

The SandS non-branded solution consists in finding a way to connect devices with proprietary hardware and software to the SansS infrastructure. The way this is achieved is via a hardware serial port offered by the appliances for the SandS motherboard to connect to them.

This document is a user manual explaining the communication protocol proposed by Gorenje and implemented as a software library for the SandS motherboard by Arduino.

2 Communication

The communication goes through 3.3V UART port on the HomeChef logic circuit. The configuration of the UART port is 19200/8/N/1 with no handshaking.

The HomeChef sends a status frame every second through UART. This status frame is also used for acknowledge of a received control frame.

There exists a second variation for the HomeChef's protocol that has been designed exclusively for running the stress tests described in D2.3 and D4.2. There are slight differences between the protocol described in this document and the one used in those tests. Refer to those deliverables for further information on the matter.

2.1 Pinout

On the HomeChef oven there is a 10 pin connector on which these pins are important:

- 3. pin: RX - receive
- 5. pin: TX - transmit
- 9. pin: GND - ground

3 Frame structure

Information is sent/received to/from the oven in a specific package format depicted in the following graphic.



image 1: data package for the HomeChef oven

- SOF
 - Start of frame byte (value: 0xFD).
- FT
 - Frame type byte (value: 0x0A).
- DATA
 - Series of data bytes.
- CS
 - Checksum byte (value: 0x0A).
- EOF
 - End of frame byte (value: 0xFE)

The frame type must be the same value as described above. The HomeChef does not support any other type for now. Also the checksum value is ignored for now, because the software is in the prototype stage.

4 Functional profile

There are now four supported functions implemented on the oven, described below.

4.1 Status function

The HomeChef oven sends info of its status through status frame.

Status data consists of a vertical bar (0x7C) delimited string containing seven substrings:

“SUBSTRING1 | SUBSTRING2 | SUBSTRING3 | SUBSTRING4
| SUBSTRING5 | SUBSTRING6 | SUBSTRING7”

Each one of the strings describes different aspects of the oven's functionality:

- SUBSTRING1
 - oven light status
 - “0” – light off,

- “1” – light on.
- SUBSTRING2
 - appliance state
 - “0” – oven idle,
 - “1” – oven running,
 - “2” – oven set and ready to start.
- SUBSTRING3
 - recipe name
 - eg.: "pizza" – selected recipe on oven
 - When no recipe is selected the string will be “blank”.
- SUBSTRING4
 - oven temperature
- SUBSTRING5
 - minutes elapsed
- SUBSTRING6
 - minutes remaining
- SUBSTRING7
 - temperature set-point

Note that NONE of the substrings are null terminated.

Example:

- “0|1|pizza|165|5|25|210”
- Light is off,
 - oven state is running,
 - selected recipe’s name is “pizza”,
 - current oven temperature is 165°C,
 - oven is 5 minutes into program with 25 minutes remaining,
 - program’s cooking temperature is 210°C.

4.2 Internal light switching

This function is a command function that the HomeChef oven receives and acts upon. The string described below must be packed in the frame structure described on top of this document.

“SUBSTRING1”

- “L0”
 - request oven light off.
- “L1”
 - request oven light on.

All substrings are NOT null terminated!

Example:

“L0”

- Light turn off.

4.3 Operation control

This function is a command function that the HomeChef oven receives and acts upon. The string described below must be packed in the frame structure described on top of this document.

“SUBSTRING1”

- “C0”
 - Stop cooking program.
- “C1”
 - Start cooking program.

Note: Start cooking command is recognized only when appliance state in status frame 1 is set to “2” (ready to start). Stop cooking command is recognized only when appliance state in status frame 1 is set to “1” (running).

All substrings are NOT null terminated!

Example:

“C0”

- Stop cooking program.

4.4 Recipe upload



image 2: frame format for recipe uploading to the HomeChef oven

- HEATER MODE

Valid values from 0x80 to 0x8B, each representing a specific heater mode:

- 0x80: upper + lower heater
- 0x81: hot fan heater
- 0x82: lower + hot fan heater
- 0x83: infra + fan heater
- 0x84: infra heater
- 0x85: lower + fan heater
- 0x86: lower heater
- 0x87: upper heater
- 0x88: fan heater
- 0x89: upper + infra + fan heater

- 0x8A: upper + infra heater
 - 0x8B: upper + hot fan heater
- TEMPERATURE
 - Numeric string with leading zeroes, defining temperature setpoint.
 - Example: “180” --> 180°C
 - Three bytes reserved.
 - All substrings are NOT null terminated!
 - DURATION
 - Numeric string with leading zeroes, defining cooking duration in minutes.
 - Example: “090” -> 1h30min
 - All substrings are NOT null terminated!

Example:

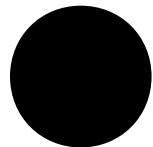
“U/x80190045” → “U” + “/x80”+ “190”+ “045”

- upper and lower heater
- 190°C
- 45 minutes

Document Information

Project reference

317947, FP7-ICT-2011-8



Context

Deliverable 2.2, workpackage 2

Document title

D2-2_Getting-Started-Guide_SandS_cross_compiler

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

9. Mar. 2014

D2.2 The SandS cross compiler (ARD + LBL)

Table of Contents

Document Information.....	1
D2.2 The SandS cross compiler (ARD + LBL).....	2
1 Introduction.....	2
2 Getting started guide: cross compiling for the SandS motherboard.....	2
2.1 Step 1: get the right computer.....	3
2.2 Step 2: install packages on the original Linux distribution.....	3
2.3 Step 3: download OpenWrt's source from git repo.....	3
2.4 Step 4: update OpenWrt's packages.....	3
2.5 Step 5 : prepare the build of OpenWrt.....	4
2.6 Step 6. build the environment.....	4
2.7 Step 7: copy libraries and .h files from target-mips to toolchain-mips.....	4
2.8 Step 8: installing libmosquitto.....	5
2.9 Step 9: compile using the toolchain.....	5

1 Introduction

The SandS hardware ecosystem refers to the set of hardware blocks, specifically designed for the “Social and Smart” project, that can be used to intervene any kind of home appliances.

The SandS motherboard is a microcontroller board with two cores, one of them running a flavor of the free and open source operating system (OS) OpenWrt.

This document is a user manual explaining how to set up a framework to compile code that can later be uploaded to the SandS motherboard via WiFi.

2 Getting started guide: cross compiling for the SandS motherboard

The SandS motherboard is basically the equivalent to an Arduino Mega board (one of the open source hardware boards designed by ARD) adding the capability of connecting to the network via WiFi. In order to achieve that, ARD has included a module called DogStamp. This module includes a processor able of running a lightweight version of the Linux OS called OpenWrt.

Even if OpenWrt is a full OS, in the sense that includes tools that allow creating software and compiling binary files from a device running the OS, it is much faster to create software for it on a different -faster-machine. This operation is known as cross-compilation.

In theory, from the point of view that the tools are all open source, it is possible to create a cross compiler for the OpenWRT OS on top of any other architecture. In practice, it is much more simple to do it on top of a Linux machine, mostly because there is much more documentation on how to do so on Linux than on any other OS.

Therefore, here we are going to see how to prepare the Software Development Environment (SDK) for OpenWrt running on a Linux 64 machine. Please be aware the instructions here noted will not work for a different OS.

2.1 Step 1: get the right computer

As mentioned earlier, the first step is having a PC running Linux 64 bits. At the time of writing, all the parties in the project involved in software production for the SandS motherboard (ARD, LBL, UNIMI, GOR) had a machine running Ubuntu 64b version 13.10 or equivalent.

2.2 Step 2: install packages on the original Linux distribution

You need to install a whole series of packages on your Linux distribution, open a terminal window and issue the following commands:

```
sudo apt-get update  
sudo apt-get install build-essential  
sudo apt-get install subversion  
sudo apt-get install git-core  
sudo apt-get install patch  
sudo apt-get install bzip2  
sudo apt-get install flex  
sudo apt-get install bison  
sudo apt-get install autoconf  
sudo apt-get install gettext  
sudo apt-get install unzip  
sudo apt-get install libncurses5-dev  
sudo apt-get install ncurses-term  
sudo apt-get install zlib1g-dev  
sudo apt-get install gawk  
sudo apt-get install libbz-dev  
sudo apt-get install libssl-dev
```

2.3 Step 3: download OpenWrt's source from git repo

Create a folder where to clone the repository containing the latest version of OpenWrt:

```
mkdir sands  
cd sands
```

Clone the online repository for the OS into a subfolder called sands-toolchain:

```
git clone git://git.openwrt.org/openwrt.git sands-toolchain  
cd sands-toolchain
```

2.4 Step 4: update OpenWrt's packages

The scripts folder inside the sands-toolchain directory contains a series of scripts that will help you updating the list of packages and installing the latest ones.

```
./scripts/feeds update -a  
./scripts/feeds install -a
```

2.5 Step 5 : prepare the build of OpenWrt

You need to run the following commands to make sure the different configuration files are created prior to compilation:

```
make defconfig  
make prereq  
make menuconfig
```

After running the last command, *make menuconfig*, a graphic user interface will appear. This is the packages selection UI. Many of the packages are included by default, however you need to add the following:

- from the Base system
 - libpthread
 - librt
 - libstdcpp
- from the Libraries
 - SSL -> libopenssl
 - libmosquitto

2.6 Step 6. build the environment

This is a pretty time consuming step that will take several hours, simply run the following and wait:

```
make -j V=99
```

2.7 Step 7: copy libraries and .h files from target-mips to toolchain-mips

The compilation will create a folder called “staging_dir” that will contain the cross-compilation tool. Now you need to copy all the files corresponding to the architecture of the processor on the DogStamp (MPIS) to the right location. Alternatively you could make links of all these files.

```
cp -a staging_dir/target-mips_34kc_uClibc-0.9.33.2/usr/lib/*  
staging_dir/toolchain-mips_34kc_gcc-4.6-linaro_uClibc-0.9.33.2/usr/lib  
  
cp -a staging_dir/target-mips_34kc_uClibc-0.9.33.2/usr/include/*  
staging_dir/toolchain-mips_34kc_gcc-4.6-linaro_uClibc-0.9.33.2/usr/include
```

Now we can cross-compile for the SandS motherboard (or Arduino YUN).

2.8 Step 8: installing libsmosquitto

As in the SandS communication protocol between the SandS motherboard and the infrastructure we are using MQTT, we are using libmosquitto, a library that contains all of the needed functions to encapsulate and parse information in the form of MQTT packages. This means that our SandS motherboard needs to have that specific library installed.

There are two possible ways to include this library: through the OpenWrt package manager -opkg- or by uploading the compiled library to the board. The default installation of the SandS motherboard doesn't come with either opkg nor libmosquitto.

If opkg packet manager is available on the board libmosquitto can be installed issuing the following commands while in an ssh session into the SandS motherboard as follows:

```
opkg update  
opkg install libmosquitto
```

However, this is not possible in the default configuration of the SandS motherboard. Please check the paragraph titled "Uploading binary files to the SandS motherboard" later in this document to get an introduction to how to copy files to the motherboard.

2.9 Step 9: compile using the toolchain

OpenWrt Cross-compile tools should now be available inside the staging directory at:

```
sands/sands-toolchain/staging_dir/
```

All of the binaries for the cross-compilation process are under:

```
sands/sands-toolchain/staging_dir/toolchain-mips_34kc_gcc-4.6-linaro_uClibc-0.9.33.2/bin/mips-op
```

The GCC compiler needed to run the compilation can be found at:

```
sands/sands-toolchain/staging_dir/toolchain-mips_34kc_gcc-4.6-linaro_uClibc-0.9.33.2/bin/mips-op  
enwrt-linux-uclibc-gcc
```

You can check which is your version of the compiler by calling:

```
sands/sands-toolchain/staging_dir/toolchain-mips_34kc_gcc-4.6-linaro_uClibc-0.9.33.2/bin/mips-op  
enwrt-linux-uclibc-gcc --version
```

In order to compile you need to create the environment variable STAGING_DIR:

```
export STAGING_DIR=/home/<directory>/sands-toolchain/staging_dir/
```

Now you can run the compiler. Simple put a simple "helloworld.c" file inside the folder where you have all of the binaries:

```
sands/sands-toolchain/staging_dir/toolchain-mips_34kc_gcc-4.6-linaro_uClibc-0.9.33.2/bin/
```

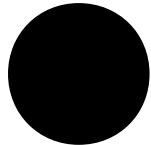
and call the compiler to create your executable:

```
sands/sands-toolchain/staging_dir/toolchain-mips_34kc_gcc-4.6-linaro_uClibc-0.9.33.2/bin/mips-op  
enwrt-linux-uclibc-gcc helloworld.c -o helloworld
```

Document Information

Project reference

317947, FP7-ICT-2011-8



Context

Deliverable 2.2, workpackage 2

Document title

D2-2_Getting-Started-Guide_SandS_I2C_configuration_tool

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

9. Mar. 2014

D2.2 The SandS I2C configurator (ARD)

Table of Contents

Document Information.....	1
D2.2 The SandS I2C configurator (ARD).....	2
1 Introduction.....	2
2 Modules' general description.....	3
3 Getting started guide: I2C Configuration tool.....	3
3.1 Step 1: connect the system.....	3
3.2 Step 2: running the configurator.....	4
3.3 Step 3: using the configurator.....	5
4 List of commands.....	5
4.1 enum – enumerate and list all connected devices.....	5
4.2 target – target a device, either based on its ID or its location in the connection chain.....	6
4.3 setid – change the ID of the device.....	6
4.4 set – set the targeted devices output.....	7
4.5 get – perform a continuous readout from the targeted device.....	7
4.6 config – perform configuration manipulation.....	8
5 Exception: The UI module.....	8

1 Introduction

The SandS HW ecosystem refers to the set of hardware blocks, specifically designed for the “Social and Smart” project, that can be used to intervene any kind of home appliances. The SandS hardware ecosystem introduces a series of modules that can communicate via a modified I2C protocol towards the SandS motherboard. The modules are, in essence, reprogrammable microcontroller boards fully addressable over I2C and come with a default firmware controlling both the communication aspects of the modified I2C as well as whatever function the module is performing.

All of the SandS modules come preprogrammed with the address 0x00 in the EEPROM of their on-board microcontroller and it is possible to reconfigure their I2C address making use of the software tool described in this document.

Please note that this applies to the modules: Relay, Mosfet, Input and Consumption. The UI module communicates through a serial port back to the SandS motherboard as it needs a dedicated communication port.

For more information about the SandS modules' hardware, please refer to the specific module's datasheet provided as part of D2.1. For more information about the SandS modules' communication protocol, please refer to the document 201309_Arduino_Sands_Deliverable_2-1_TWI-protocol.pdf also part of D2.1.

This document is looking into a software tool called “The configurator” used to change the default address on the SandS modules above mentioned. It is possible to build a chain of as many as 127 modules of different kinds and let this tool configure the addresses automatically. Once program is done iterating throughout all the modules in the chain, they will have got a new ID that will remain in the modules' respective EEPROMs. Therefore it won't be necessary to run this software again, unless the modules are about to be reused in a different scenario.

2 Modules' general description

All of the modules have a very similar structure at the communication level as shown in the following diagram.

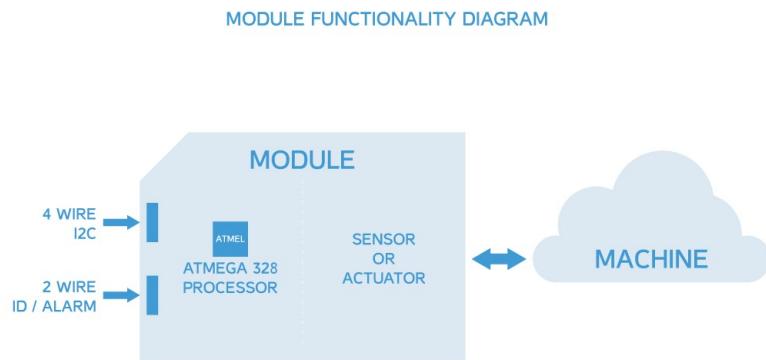


image 1: module functionality diagram, or overview of the way a module works

For a more in-depth explanation about the pinout of the different connectors depicted on the diagram, please refer to the individual datasheets for each module. It is possible to program each one of the modules directly from a computer using a USB-Serial cable and the Arduino IDE with the SandS libraries. The source code provided as part of D2.1 includes the basic firmware for each one of the modules. However it is not recommended to modify it as we cannot guarantee the same functionality as the one described here.

3 Getting started guide: I2C Configuration tool

The SandS Configurator (SC) is a simple command line tool (CLI) to configure the I2C unique address number of the different modules used in a certain installation. This tool is provided as part of the deliverable D2.2 in the form of source code that can be compiled using standard GCC in any operating system. As a proof of concept, the Windows binary is also provided.

For the SC to be able of exchange data with the modules, they need to be connected to either a SandS motherboard or an Arduino UNO or Arduino YUN using the SandS shield designed as a test tool for T2.1. For more information about this shield, please refer to ARD's IPR for the SandS project, where it is possible to find more information about all of the different hardware modules, boards, and design iterations created for the SandS project.

3.1 Step 1: connect the system

First thing you need to do is connecting all the electronic components and upload the software needed (using the Arduino IDE) for the motherboard or Arduino equivalent to communicate back to the configurator tool:

1. Connect all modules to the shield or motherboard.
2. Besides the normal 4-wires I2C cable you will need to connect all of the modules using the extra ID/Alarm 2-wires cable in order to arbitrate the communication while assigning the IDs to the modules. Remember as well to power the SandS motherboard using a 12VDC power supply.

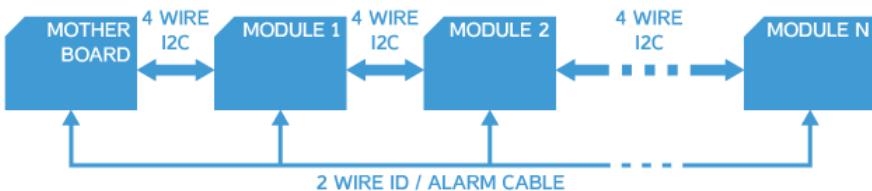


image 2: daisy chained modules using the 4 wire I2C and the extra 2 wire ID/Alarm cable

3. Download the configurator sketch to the board of choice.
Make sure you have selected Arduino Mega 2560 if you are using the motherboard, or the Arduino Leonardo/Yún/UNO if you are using the shield.

3.2 Step 2: running the configurator

Before proceeding with this step, you should know which is the serial port your board is connected to.

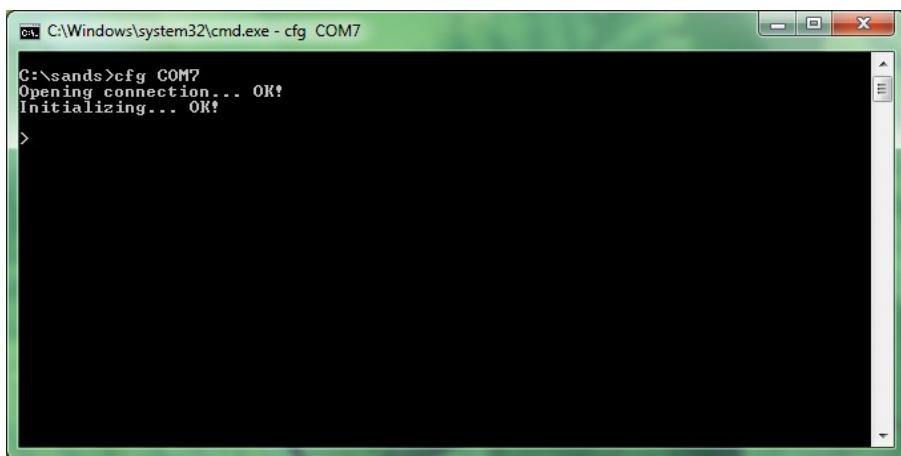


image 3: launching the configurator

On the command line, run the configurator with the port or device name as an argument:

- for Windows: cfg COM*
- for Linux: cfg /dev/ttyACM*
- for Mac: cfg /dev/tty.usbmodem*

Where the asterisk represents the unique identification number for the USB port on which your board is connected.

3.3 Step 3: using the configurator

Once the program is running, type help for a list of commands the tool can perform:



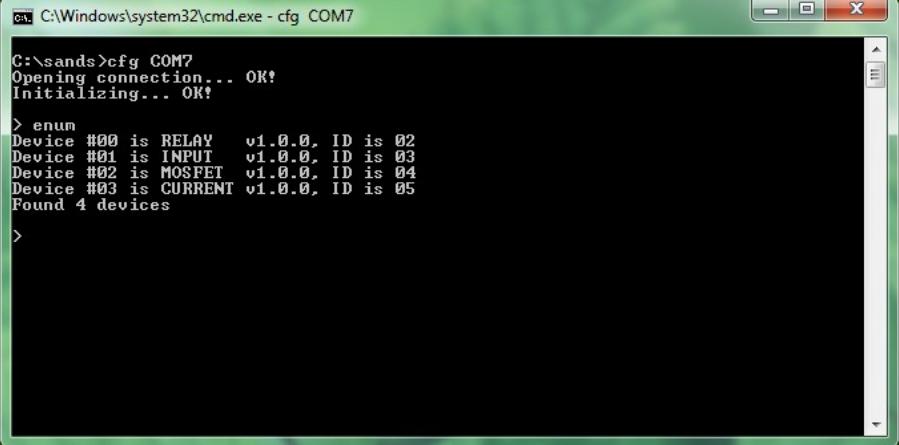
```
C:\sands>cfg COM7
Opening connection... OK!
Initializing... OK!

> help
Available commands:
enum    Enumerate all connected devices
target   Target device
setid   Program new ID
config   Configuration manipulation
get     Perform readout
set     Set output
>
```

image 4: configurator's inline help

4 List of commands

4.1 enum – enumerate and list all connected devices



```
C:\sands>cfg COM7
Opening connection... OK!
Initializing... OK!

> enum
Device #00 is RELAY v1.0.0. ID is 02
Device #01 is INPUT v1.0.0. ID is 03
Device #02 is MOSFET v1.0.0. ID is 04
Device #03 is CURRENT v1.0.0. ID is 05
Found 4 devices
>
```

image 5: enumeration of devices

This command is useful for initial configuration and will list devices based on the order of the connection chain.

4.2 target – target a device, either based on its ID or its location in the connection chain

To target by ID simply use: target hh (where hh is the ID, hexadecimal)

To target by location use: target #hh (where hh is the location, hexadecimal)

```
C:\sands>cfg COM7
Opening connection... OK!
Initializing... OK!

> enum
Device #00 is RELAY v1.0.0, ID is 02
Device #01 is INPUT v1.0.0, ID is 03
Device #02 is MOSFET v1.0.0, ID is 04
Device #03 is CURRENT v1.0.0, ID is 05
Found 4 devices

> target 02
Targetting RELAY v1.0.0, ID is 02
> target #04
Targetting... #04
Targetting MOSFET v1.0.0, ID is 04
>
```

image 6: choose a device to perform operations with

Once a device is targeted it is possible to manipulate the device by using the following commands: setid, set, get, and config.

4.3 setid – change the ID of the device

In the following example we will change the ID of the RELAY board from 02 to 06:

```
C:\sands>cfg COM7
Opening connection... OK!
Initializing... OK!

> enum
Device #00 is RELAY v1.0.0, ID is 02
Device #01 is INPUT v1.0.0, ID is 03
Device #02 is MOSFET v1.0.0, ID is 04
Device #03 is CURRENT v1.0.0, ID is 05
Found 4 devices

> target 02
Targetting RELAY v1.0.0, ID is 02
> setid 06
> enum
Device #00 is RELAY v1.0.0, ID is 06
Device #01 is INPUT v1.0.0, ID is 03
Device #02 is MOSFET v1.0.0, ID is 04
Device #03 is CURRENT v1.0.0, ID is 05
Found 4 devices

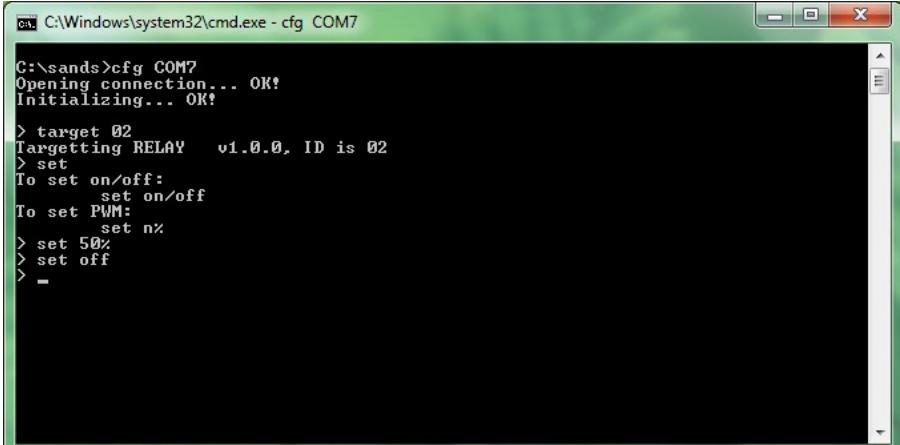
> =
```

image 7: change a device's ID

The device needs an ID for easy addressing by the implementation.

4.4 set – set the targeted devices output

Simply typing set with no argument will display options for the targetted device, as illustrated below:

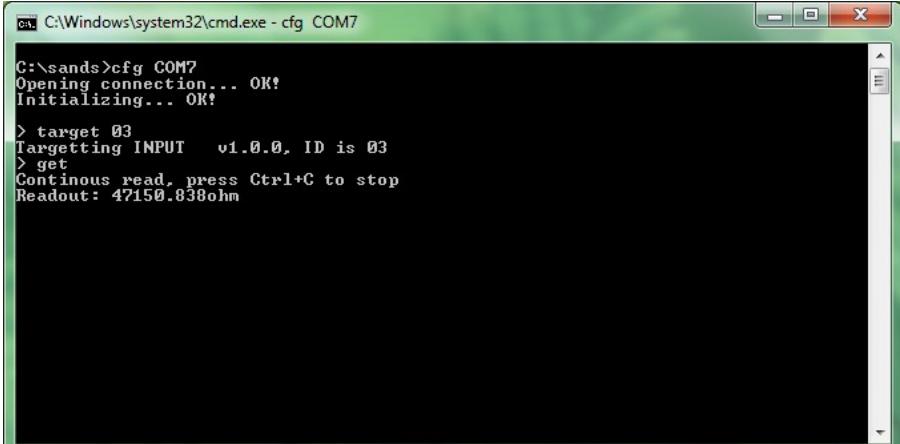


```
C:\sands>cfg COM7
Opening connection... OK!
Initializing... OK!
> target 02
Targetting RELAY v1.0.0, ID is 02
> set
To set on/off:
    set on/off
To set PWM:
    set nx
> set 50%
> set off
> -
```

image 8: set a device's output

4.5 get – perform a continuous readout from the targeted device

The readout will be made using current configuration options as illustrated below:



```
C:\sands>cfg COM7
Opening connection... OK!
Initializing... OK!
> target 03
Targetting INPUT v1.0.0, ID is 03
> get
Continuous read, press Ctrl+C to stop
Readout: 47150.838ohm
```

image 9: reading values from an input module

This is useful for testing your current configuration, especially for the INPUT board which has many configuration options.

Once you are done, press Ctrl+C to cancel the readout and return to the prompt.

4.6 config – perform configuration manipulation

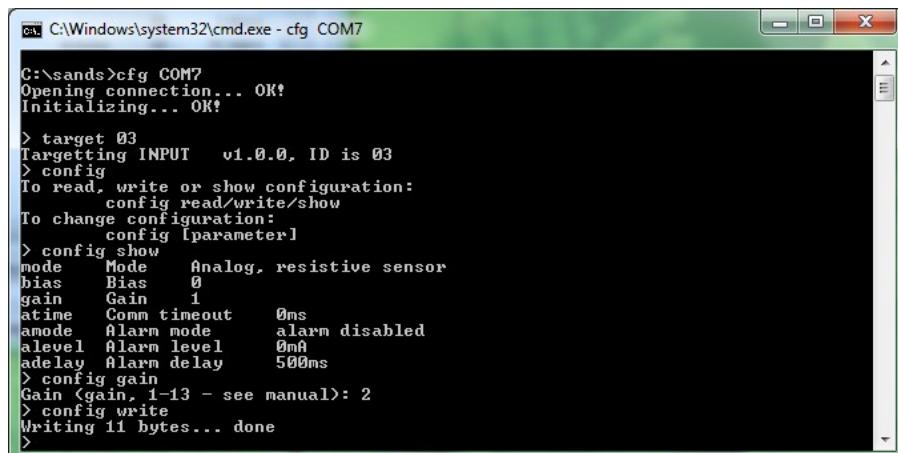
This is used to read, display, modify and write configurations to the targeted device.

- config read: read the current configuration (not that this is automatically done when targeting)
- config show: show all configuration options as well as their current setting
- config write: write configuration to the device after changes are made. Note that this operation must be done prior to testing using get

To manipulate a value, use config value (where value is a valid parameter, see config show).

This will bring up a list/range of available settings as well as a prompt for the new value.

The process is illustrated below:



```
C:\>sands>cfg COM7
Opening connection... OK!
Initializing... OK!

> target 03
Targetting INPUT v1.0.0, ID is 03
> config
To read, write or show configuration:
    config read/write/show
To change configuration:
    config [parameter]
> config show
mode Mode Analog, resistive sensor
bias Bias 0
gain Gain 1
atime Comm timeout 0ms
amode Alarm mode alarm disabled
alevel Alarm level 0mA
adelay Alarm delay 500ms
> config gain
Gain <gain, 1-13 - see manual>: 2
> config write
Writing 11 bytes... done
>
```

image 10: using the config command

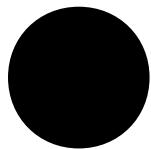
You can find a list of all parameters and their exact explanations in the protocol reference manual in the document *201309_Arduino_Sands_Deliverable_2-1_TWI-protocol.pdf* which is part of deliverable D2.1.

5 Exception: The UI module

The UI module is a different type of module, due to it's specific needs in terms of speed of access to it from the SandS motherboard. It doesn't use the I2C protocol as a way to communicate but a 10 pin header for a flat cable. This cable provides bi-directional UART communication to the motherboard as well as several GPIO lines.

The UI board can be programmed using a standard AVR ISP connector, or as an Arduino Leonardo device using the USB connector at the top of the board.

Document Information



Project reference

317947, FP7-ICT-2011-8

Context

Deliverable 2.2, workpackage 2

Document title

D2-2_Getting-Starter-Guide_Sands_motherboard

Company information

Arduino SA

Via D'Alberti, 1

6830 Chiasso

Switzerland

Document version number

v0001

Author

David Cuartielles

Contact information

d.cuartielles@arduino.cc

Document created

1. Aug. 2013

Last modification

9. Mar. 2014

D2.2 The SandS motherboard (ARD)

Table of Contents

Document Information.....	1
D2.2 The SandS motherboard (ARD).....	2
1 Introduction.....	2
2 Getting started guide: The SandS motherboard.....	2
2.1 Adding modules to the motherboard.....	3
3 Board general description.....	3
3.1 Basic functionality.....	4
4 IDE.....	4
4.1 Installation.....	5
4.1.1 Windows.....	5
4.1.2 Linux.....	5
4.1.3 Mac OSX.....	5
4.2 Menus and toolbars.....	5
4.3 Powering the board.....	6
4.4 Uploading your first sketch.....	7
4.4.1 Pin equivalences to Arduino Mega.....	7
4.4.2 Sketchbook: storing your sketches.....	8
4.5 Libraries.....	9
4.6 Code examples.....	9
4.7 Motherboard's WiFi connectivity.....	10

1 Introduction

The SandS hardware ecosystem refers to the set of hardware blocks, specifically designed for the “Social and Smart” project, that can be used to intervene any kind of home appliances. This document is a user manual explaining how to get started using the SandS motherboard and how to write software for its AVR core.

2 Getting started guide: The SandS motherboard

The SandS hardware ecosystem is made of a motherboard and a series of modules aimed at externally controlling any kind of home appliance available in the contemporary home. We distinguish two main methods of controlling an appliance:

- by directly hacking any of the sensors and actuators in the appliance
- by connecting to a serial port connector in the appliance

The SandS motherboard is an Arduino compatible board that offers both methods of connectivity. It comes with a WiFi module to hook up to the Internet via an existing residential Internet connection.

The SandS motherboard can be programmed from a computer using the Arduino IDE and a special library dedicated to handle the communication to the different modules as well as using the special

HomeChef library to connect to Gorenje's oven through a serial port.

2.1 Adding modules to the motherboard

The modules can communicate with the motherboard either using a Two Wire Interface (also known as I2C) or a serial port. The first method allows connecting up to 127 different modules, no matter whether they are sensors, actuators or a combination of both, to the motherboard. The TWI can address 127 different devices. The motherboard will act as communication master and the different modules as slaves.

The second communication method -the serial port- has the limitation of being a one-to-one link, in other words, only one module at the time can make use of that connection. The motherboard has 4 hardware serial ports: one is used for reprogramming the board, a second one is used to communicate to the WiFi module, the third is hooked up to a serial port controller chip that can connect to both 485 and 232 serial ports, the final port is meant to be used to connect to the SandS UI module.

Therefore, the communication between modules and the motherboard using a serial port, is not recommended for standard operation, but it is possible for debugging the modules. It is also possible to connect any module directly to a computer using a USB-serial cable, except for the UI module that runs on a USB native processor (the ATMega 32u4) and therefore it doesn't need any external USB-serial converter.

3 Board general description

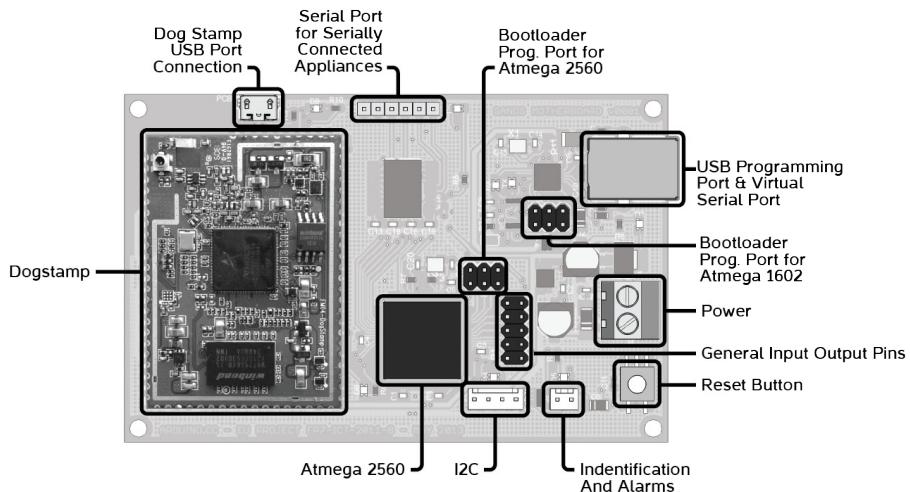


image 1: block description, motherboard; note, revision 4 of the mother board and later have removed the micro USB port connection on the top area of the board

The SandS motherboard can be understood as a two-core microcontroller board where one of the processors is dedicated entirely to the communication over WiFi and runs a trimmed down version of the Linux operating system, while the second processor is used to program specific drivers for controlling the appliances.

The following diagram explains how the motherboard works internally:

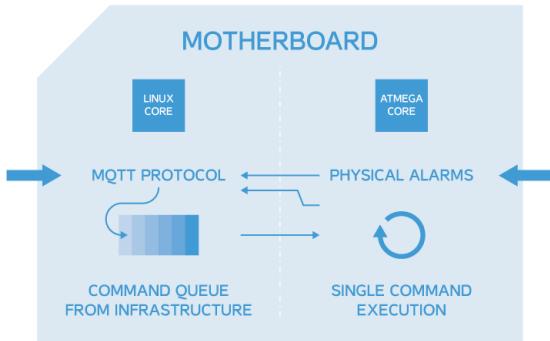


image 2: diagram showing the two cores in the motherboard and how they exchange data

3.1 Basic functionality

The SandS motherboard will connect to the SandS infrastructure once its network interface has been properly configured. Then it will queue commands arriving over WiFi into the Linux processor memory and will serve them one by one to the AVR core.

The AVR core will communicate either to each one of the SandS modules used in the appliance, or directly through a serial port in case the appliance offers that possibility.

In case there is an alarm coming from either the network or any of the modules, the AVR core will be taking care of interrupting the operations at the physical level and reporting back to the SandS infrastructure.

4 IDE

The Arduino IDE is constantly evolving, at the time of writing this guide there were three different versions of the official Arduino IDE:

- Arduino 1.0.5: the stable branch, supporting only boards using Atmel processors from the ATMega family, will reach obsolescence during the Spring 2014
- Arduino 1.5.6-r2: the development branch, running both ATMega processors and ARM core processors, like the Arduino Due. This IDE supports as well the new Arduino Yun boards that can be reprogrammed wirelessly
- Arduino 1.5.X for Galileo boards: a specific flavor of the Arduino 1.5.3 IDE designed to run on the Intel Galileo boards, using Intel's Quark processor family. This will be merged with the Arduino 1.5.3 (and subsequent) IDE

In order to make the process of getting started with the SandS boards as smooth as possible, we

have decided to stick to the latest version of the stable branch (1.0.X) which will require the users of the motherboard to include the different software libraries and examples generated throughout the project.

The software provided for the project has not been tested on the development branch of the Arduino IDE.

4.1 Installation

The Arduino IDE comes as a compressed file (ZIP for Windows, TAR-GZ for Linux, DMG for Mac), you need to follow the installation instructions for each operating system as described on the Arduino website at: <http://arduino.cc/en/Guide/HomePage>

Keep in mind that the SandS motherboard is hardware equivalent to the Arduino Mega 2560 and therefore you will need to configure it like that on the board's menu.

4.1.1 Windows

Also remember that the serial ports are represented differently in different operating systems. For Windows, your SandS motherboard will show up as a COM* port, where * will represent a certain number. You can see where your board is connected by checking Window's device manager.

You should remember installing the drivers properly when connecting the board for the first time. This also means that you will need to power up your board properly with an external power supply.

4.1.2 Linux

For Linux the operating the SandS motherboard shows up at /dev/ttyACM*. If it is the only Arduino compatible board connected at that time to your computer, it will most likely take the name /dev/ttyACM0. You should check this by checking which are the ones showing up on the tools → serial port menu.

You will require no drivers.

4.1.3 Mac OSX

On Mac OSX the Arduino firmware makes the serial port show up as a USB modem device. Therefore the serial port address is at /dev/cu.usbmodem*, where the * is a unique identifier for that board.

You will require no drivers.

4.2 Menus and toolbars

The Arduino IDE has a series of menus and one single toolbar offering shortcuts to the most used options in those menus.

You can check those functions on the Arduino website, however, as a reminder, here the list of possible commands:



check whether your program will work



upload the program to the Arduino board

-  create a new program
-  open a program
-  store the program in the computer's hard-drive
-  (all the way to the right) open a serial port monitor to the board

4.3 Powering the board

In order to get the board to work, you need an external power supply. It is recommended to use a 12VDC supply able of providing up to 2A of current. That will be enough for almost any configuration of SandS motherboard and SandS modules you could prepare to control any kind of appliance.

The power sequence of the board requires that you first power up the board and then you connect the USB cable to the computer. The polarity of the power connector on the motherboard can be seen in the following diagram:

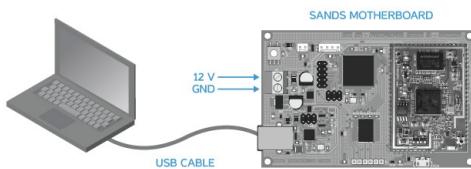


image 3: connecting the SandS motherboard to the Arduino IDE, first power the board, then connect the USB cable

4.4 Uploading your first sketch

In order to simply test that the communication between your PC and your motherboard works, you can try to upload a simple sketch to the motherboard. The following program will make the LED labeled as D5 blink. If you are able of replicating this experiment it will mean that you will have installed the drivers for your board properly.

```
int led = 37; // the LED D corresponds to Arduino's pin 37

void setup() {
    pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
    delay(1000);
    digitalWrite(led, LOW);
    delay(1000);
}
```

4.4.1 Pin equivalences to Arduino Mega

The SandS motherboard is hardware equivalent to the Arduino Mega 2560. It has a more limited collection of pins. Follows a table depicting the connections between the pins on the Arduino Mega and the pins on the SandS motherboard. This will allow you accessing them directly from the Arduino IDE.

Pin on SandS motherboard	Digital pin on Arduino Mega	Analog pin on Arduino Mega
Alarm	36	A8
Identification	37	A9
SDA	20	-
SCL	21	-
MB_RXD/D3	17	-
MB_GPIO3/D7	9	-
MB_GPIO1/D5	7	-
MB_TXD/D2	16	-
MB_GPIO0/D4	6	-
MB_GPIO2/D6	8	-

table 1: comparison between pins on the Arduino Mega and the SandS motherboard

If you want to localize the pins on the motherboard, use the following image:

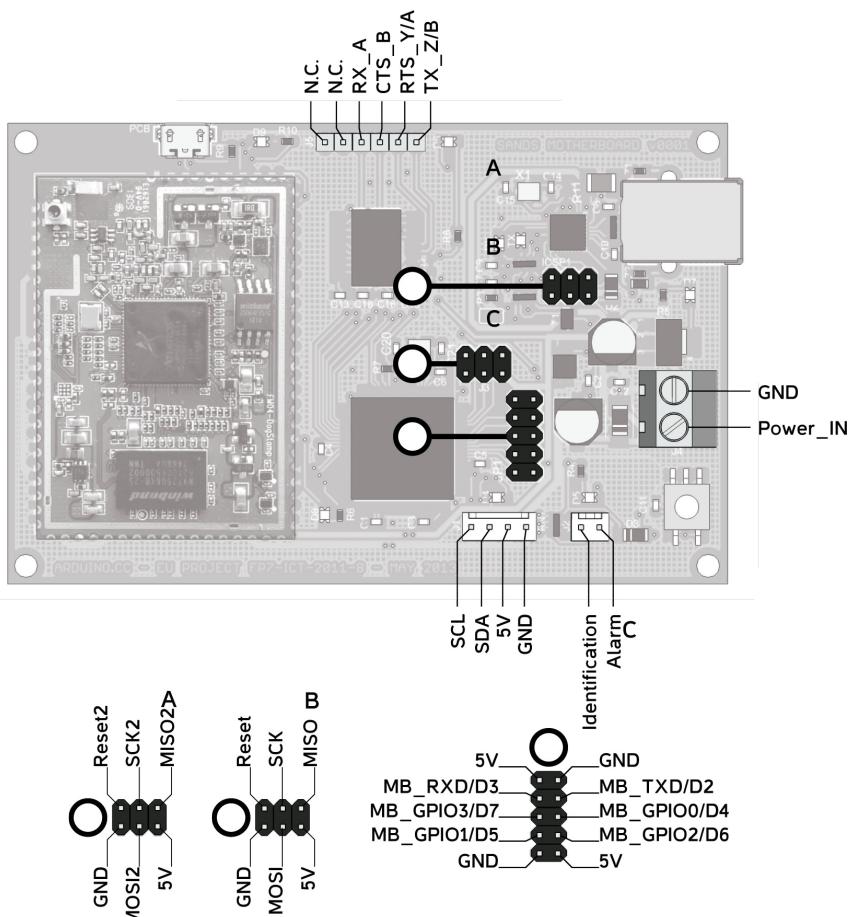


Image 2: pinout for the Sands Motherboard

4.4.2 Sketchbook: storing your sketches

If you want to store your programs, you can save them anywhere in your hard-drive or on an external flash drive. However, if you don't choose a folder where to store the files, by default they will end up in your "Documents" folder (also known as "My Documents" in Windows) in a subfolder called "Arduino".

You can always call back the programs you saved earlier by choosing "open" from the menu or toolbar and navigating to wherever it is you save them. If you stored them in the default folder, you will find them easily under the menu "File → Sketchbook".

4.5 Libraries

Arduino has created two different libraries that need to be imported in most of the examples used for communicating with either modules or third party proprietary devices like the HomeChef by Gorenje.

The libraries can be found as part of the IPR generated by Arduino for the SandS project and are protected under a GPL license.

If you want to install the libraries, do the following:

1. close the Arduino IDE if you had it open
2. unzip the Arduino IPR Sands file inside your computer, at the time of writing the latest version of this file was:
`Arduino_IPR_2013-2014.zip`
3. you should go to the default folder where Arduino stores your programs, also known as Sketchbook and create a sub-directory called "libraries" in case it doesn't exist
4. from the unzipped files, copy the whole content of the sub-directory:
`/Arduino_IPR_2013-2014/SW/SandS_motherboard/libraries/`
into your newly created libraries folder
5. you should end up having two new folders inside the libraries one:
 1. HomeChef – this is the library used for the communication with Gorenje's oven
 2. SandS – this is the library used to communicate with the different modules

4.6 Code examples

As part of the same compressed file mentioned earlier, there are a series of examples that can be used for performing the different tasks required in the project. They can be found under the directory:

`/Arduino_IPR_2013-2014/SW/SandS_motherboard/sketches/`

In there we have categorized the different software packages created for the project into:

- configuration: contains only the `SandS_configurator` sketch, used to program the ID number on the different boards in the system
- host: contains two different programs:
 - `SandS_host`: used to communicate to the DogStamp via serial monitor from a PC
 - `SandS_gorenje`: used to test the HomeChef firmware from the SandS motherboard
- modules: contains the firmware for the different modules
- projects: contains the software examples used with the coffee machine prototype shown in Brussels in the Spring 2013
- under_development: different sketches still under test, among others, some basic tests for the SandS UI module
-

4.7 Motherboard's WiFi connectivity

Most of the WiFi connectivity on the SandS motherboard falls under the work of a different work-package. It is worth mentioning that the motherboard cannot be re-programmed via the WiFi connection and that such a connection is only used to connect to the SandS infrastructure. Also, the mechanisms to upload new programs to the Linux core of the motherboard are explained in the document, part of this deliverable, dedicated to the cross compilation of code for the SandS motherboard, includes information about how to upload binary files to the Linux processor.



APPENDIX C, PUBLICATION 3

Appendix C, Publication 3

D. Cuartielles, E. Katterfeldt, G. Dabisias & A. Berner (2015).
Delivery number 4.2: Report on Final STEM Learning Kit with Integrated Learning Analytics for Trials (Malmö: PELARS, 2015) [PELARS].

DELIVERABLE IDENTIFICATION SHEET

IST Project No.	FP7 – Technology Enhanced Learning - 619738
Acronym	PELARS
Full title	Practice-based Experiential Learning Analytics Research And Support
Project URL	http://www.pelars-project.eu/
EU Project Officer	Francesca Borelli

Deliverable	D4.2 Final STEM Learning Kit with Integrated Learning Analytics for Trials	
Work package	WP4	

Date of delivery	Contractual	M19	Actual	M22 as 12-10-2015
Status	Version 1.0			
Nature	Report			
Dissemination Level	Public			

Authors (Partner)	ARD, SSSA, UB, CIID			
Responsible Author	David Cuartielles Partner	ARD	Email Phone	d.cuartielles@arduino.cc +46 763 98 58 18

Keywords	
----------	--

Version Log			
Issue Date	Rev No.	Author	Change
05-10-2015	0.1	D. Cuartielles	First Internal Version: ARD, UB, CIID, SSSA
11-10-2015	1.0	D. Cuartielles	Revised by partners: SSSA, MAH, UCL



1

LEARNING+MAKING



PELARS

**D4.2 Final STEM
Learning Kit with
Integrated Learning
Analytics for Trials**

Authors:

D. Cuartielles, E. Katterfeldt, G. Dabisias, A. Berner

Internal reviewers:

E. Ruffaldi, D. Spikol, M. Cukurova



ABOUT THIS DOCUMENT

This deliverable is an overall status report on the technical development and kit design made within the PELARS project prior to launching a series of trials with students within three different learning scenarios.

The document presents a series of tools: hardware, software and crafting materials that will both enable the students perform a series of tasks, but also get real time feedback about the state of their projects. The feedback will also be sent to the teachers to inform them about potential issues faced by the students, allowing them take actions there where it is most needed.

The tools here presented have been developed for the project based on the research made mainly as part of workpackages WP2 and WP4, but also on the technical descriptions of WP5.

When it comes to the technical development, it advanced faster than anticipated, thanks to a relocation of human resources, what allowed the project to be presented at Ars Electronica in September 2015 and fulfill some of the partners' plans for trials with students to start in late October 2015. At the time of writing 4 different series of boards have been produced adding up to more than 700 circuits of 13 different types that will be put in the hands of students during the different trials.

The electronic development platform is now called TALKOO kits (formerly known as PELARS kits) that together with the PELARS crafting materials and the PELARS visualization tool will be at the core of the trials to be performed with groups of university students in interaction design and engineering, as well as high school students in different countries. PELARS has an extensive trial plan that is described in WP7's deliverables and that builds on the results of this deliverable D4.2.

Abbreviation	Description
PR	Public Report
WP	Work Package
LA	Learning Analytics
LAS	Learning Analytics System
LMS	Learning Management Systems
Partner Abb.	Description
SSSA	Scuola Superiore Sant'Anna
MAH	Malmö Högskola (Malmoe University)
UB	University of Bremen
ARD	Arduino SA
CIID	Copenhagen Institute of Interaction Design
UCL	Institute of Education, University College London

TABLE OF CONTENTS

DELIVERABLE IDENTIFICATION SHEET.....	1
ABOUT THIS DOCUMENT	3
1 INTRODUCTION	6
2 LINKS TO OTHER DELIVERABLES	6
3 DESCRIPTION OF MATERIALS.....	6
Interaction Design Scenario: Musical Instruments	6
General description of the activity	7
Learning Activities	7
High School Scenario: Interactive Baggage Sorting Hallway.....	9
General description of the activity	9
Learning activities.....	9
Engineering Scenario: Smart Home	10
General description of the activity	10
Learning activities.....	11
4 DESCRIPTION OF KIT “LAB WARE” MATERIAL	12
Relation to Kit Learning Activities	12
The basic add-on set.....	13
Context- and task-specific material sets	14
Interaction Design material set	14
High School material set.....	16
Engineering Education material set	16
5 TALKOO KIT’S INNOVATION WITHIN EDUCATION.....	17
Hot-plugging Mechanism	17
Visual Programming	17
Mix of Embedded and PC Computation	18
Technical Solution	18
Hardware Prototypes	18
Software Development	28
Firmware Development.....	33
Visualization Tools.....	37
Connecting the Kit to the Collector	38
Manufacturing Rounds.....	39
6 CONCLUSION	40

1 INTRODUCTION

This deliverable summarizes a large portion of the work done in WP4. We have looked at how to match the educational goals from WP2, mainly from D2.2, with a series of tools that could be used as part of the educational trials. These tools are the so-called TALKOO kit, the name given to the PELARS kit for modular electronics, and the PELARS software tool for the Arduino IDE that allows to visually program the relationships between the blocks connected to the computer.

The goal of this deliverable is to contextualize the creation of TALKOO and to describe each one of the modules, what they do and how they can be used as part of the PELARS interactive setup. At the same time, we will make a revision of the different initiatives within PELARS linked to the TALKOO kit like the visualization tools for students and teachers. Those tools will be explored further in later deliverables.

2 LINKS TO OTHER DELIVERABLES

The TALKOO kit, as well as the visualization tools presented in this deliverable are built on top of the work realized in the following list of previous deliverables:

- D2.2 where different educational kits were presented in terms of reprogrammability of the components and the recommendations for the PELARS educational materials were outlined
- D4.1 where different educational kits were presented as well as the

methodology for prototyping and some preliminary ideas about the information visualization techniques

- D5.1 where we explored the software framework and selection of learning metrics and analytics
- D5.2 where the interoperability between the electronic kits (TALKOO), the furniture and the collector system for the LAS were described

3 DESCRIPTION OF MATERIALS

By materials we mean the set of parts that students will use during a trial in order to accomplish the series of tasks part of a learning activity. These parts can be the electronics kit needed to create interactive experiences, but also the crafting materials like cardboard, glue, scissors and others. The following section explores different types of materials based on the learning scenarios that are part of the PELARS trials.

As part of T4.2 and T4.3, integrated scenarios with different learning activities were developed for the three education contexts high school, interaction design post-secondary education and engineering post-secondary education. These scenarios have informed especially the design of the material sets that complement the hardware kit and IDE (see below). Scenarios build on use cases by partner ARD and were developed together with

educational partners IOE; UB, and MAH. While the integrated scenarios cover all perspectives of the PELARS environment, systems and curricular goals, in the following we focus on the description of learning activities and aspects of the scenarios that were most relevant for the kit designs.

Interaction Design Scenario: Musical Instruments

The Interaction Design (IxD) scenario is based on the idea of having students in IxD at the university level. In this case we will run trials with students at Malmö University (and potentially CIID), where it is possible to recruit both BA and MA students to test our system. Due to the fact that IxD students have a mixed technical background, it is not possible to anticipate how deep they could go into



exploring some aspects of a technology. The learning scenario implies the creation of a music instrument, what will bring the students to explore different qualities of sound and how to interact with it. This is a typical IxD experiment where the educators in the program get the students to design a system that should be both deterministic and unexpected.

General description of the activity

This PELARS educational activity is designed for interaction design (IxD) education. Students are expected to be beginners in physical computing and programming (i.e., no previous knowledge is required). The estimated duration is up to 15 hours, which can be organized as a project week, or split among during several consecutive weeks.

The IxD course is divided into labs and a longer project period. The labs are where the procedural skills are taught about physical computing and interaction. The labs are short (a 2-4 hour session fitting with students' course schedule) and the students need to complete a lab assignment. PELARS will focus on the early labs to introduce design students to physical computing hardware and software. These intro labs are straightforward lab sessions where the groups of students work on closed tasks with examples and clear outcome(s) e.g. getting the piezo speaker to make sound and the LDR to trigger events. The lab assignment gives a project to the students to create an interactive musical instrument that combines the previous actuators plus new sensors.

The overall topic is "make a musical instrument". The course comprises several learning modules: three introductory tasks and two "final" design projects that build up on each other:

- make a simple instrument (synth guitar)
- make a complex instrument (6-axis Theremin synth)

The design focus is on the design projects which allow for bringing in concepts of embodied interaction. In order to introduce the concepts of physical computing, programming and physics required for successfully implementing

the design projects, introductory tasks are performed.

As building material are provided: The PELARS Arduino kit, the IDE and crafting material (cardboard, guitar neck model, textile material). Tools from the students' lab like laser cutters can be used.

The minimal required hardware modules to be included in projects A and B are pre-given: for project A) the modules are 3 buttons, a light sensor, piezo. For project B) it is an accelerometer sensor, 2 buttons, piezo. Nevertheless, the full PELARS kit is provided to allow for extending and customizing projects later.

The Arduino modules are programmed using the Arduino IDE, which runs, together with PELARS learning system, on the computer that is part of each workstation. The students are asked to document their process with the PELARS learning system.

Learning Activities

At the first date, the students build teams of three. Each group takes place at a PELARS workstation. The teacher gives a general hands-on introduction into circuiting and programming with the PELARS Arduino kit.

Students build and program first examples that contain kit modules and programming logic they'll need for design projects A and B:

- create simple sound: Introduction into sound generation: how to generate sound with the Piezo module, how to program using frequency values and how to control the set-up with a simple digital input (button)
- make a melody: Additionally to the previous example, timer functions are introduced. This enables to create rhythms, and thus a melody.
- make a random noise generator: (programming) concepts of random values are introduced based on the previous examples. The students learn how to use random values in their programs to modify sounds

lab and sessions	project kit modules	hardware	programming / physical computing concepts	STEM and IxD concepts
intro lab 1: create simple sound	hub, piezo, button		frequency/pitch (OSC functions), digital input, if/else control structures	physics: sound pitch
intro lab 2: make a melody	hub, piezo		+ timer (control) structures, arrays	sound/music: rhythm, melody
intro lab 3: make a random noise generator	hub, piezo		+ randomness	
design project A: make a simple instrument (synth guitar)	hub, piezo, light sensor, 3 buttons		+ analog sensor input (1-dimensional), variables	design physical instrument, embodied interaction, sound design
design project B: make a complex instrument (6-axis Theremin synth)	hub, piezo, accelerometer sensor, 2 buttons		+ analog sensor input (6-dimensional)	embodied interaction (movements/wearable computing), physics: acceleration, orientation in 3d-space

Table 1: learning outcomes for the IxD students scenario

Each of the introductory tasks comprises the steps:

1. Students read exercise instructions
2. Students depict kit hardware modules, putting them physically together, connecting modules to PELARS system (IDE)
3. Students choose corresponding visual blocks on the IDE screen and connect them with node points that allow connecting them together
4. Students program the visual modules by integrating visual modules with programming logic
5. Students iteratively debug their setup
6. Meanwhile, students document their progress and outcome

After the introductory tasks, students implement the design projects A and B: the synth guitar and the 6-axis Theremin sound. For each, they circuit the technical part of the

projects and create simple programs for it (following the steps described above). They develop design ideas on their whiteboard how to integrate this into an innovative instrument with a focus on designing the user interaction with the instrument and the pre-given modules. During the following hours, students work iteratively on their design projects. This includes repetitive activities of building the hardware circuit, programming the board, testing the artefact, enhancing the artefact with additional material, debugging, refining, and documenting this process.

The students start with design project A) to get acquainted to working with a sensor module (light sensor). Using simple button modules, a light sensor module and the additional kit material, the students can create a live guitar-like instrument. The light sensor is programmed to sense coverage from a student's hands within certain values, which when strummed above the sensor in a guitar-

playing motion together with buttons held down, gives a guitar-like playing mechanism.

Then they move on to design project B) which has a stronger focus on body-sound interaction design and embodiment. Continuing from the synth guitar assignment, the students now get to use the 6-axis accelerometer sensor module. This complex sensor gives full three dimensional data from two separate sensors of movement forward, backwards, sideways, tilt and rotation. Used in conjunction with other sensors, this setup can bring very novel control attributes to sound and vision creativity and embodied interaction. Attaching the 6-axis on limbs with textile material and using mapping functions, students can invoke novel sound generation and control that is hard to find in the consumer or commercial audio professional world.

At the last date, all students give a presentation where they have to reason their design decisions.

High School Scenario: Interactive Baggage Sorting Hallway

In this case we have designed a learning scenario where we will give high school students (upper secondary students) the task of developing a system that should be capable of performing a simple task. These trials will involve, among others, students that usually take part at activities organized by CitiLab Cornellà.

General description of the activity

This PELARS educational activity is designed for high school students in STEM subjects, it explores physics and programming through a real world case for the students to design and prototype a baggage sorting machine. The activity introduces the students to physical computing, complex programming, and the principles of additive light. The estimated duration is 10-15 hours, which can be organized as a project week, or split among during several consecutive weeks.

The following building material are provided: the PELARS Arduino kit, the IDE and crafting material. Essential kit components for interactive hallway project and introductory labs comprise the hub, a color sensor, an RGB LED and LED, a button, a potentiometer. Further non-technical kit components are a hallway model, baggage paper models of different color and tools to build.

Learning activities

The learning activities could be described in the following steps:

1. the teacher presents the problem: “Build a baggage sorting prototype. Using the color sensor to determine which luggage is red, green, blue, and unidentified” and gives a general introduction about the circuiting and programing of the Arduino kit.
2. students build introductory tasks 1-4 with the help of the teacher to be familiarized with tools and programming concepts required for the final project.
3. students are grouped and each group of students is asked to plan in detail how they would use the kit provided to detect colors in an interactive hallway.
4. monitoring systems are finalized with feedback from the teachers.
5. students decide what values are optimum for color detection and program the Arduino kit to maintain these levels.
6. students plan what they will measure and how often. They design and develop output indicators for the baggage colours with the RGB LED. Their methods are evaluated in terms of control of variables strategy, reliability, and validity of data collection.

For each introductory task and the STEM project, students perform the steps with the PELARS kit, IDE and documentation system described above for interaction design.

The introductory labs built on each other. They start with simple digital output control. Then, digital input and logical structures are introduced, followed by concepts required for

lab and project sessions	kit hardware modules	programming / physical computing concepts	STEM and IxD concepts
intro lab 1: blink LED on/off with timer	hub, LED	timer (control) structures, digital output	Math/CS: binary (on/off, I/O)
intro lab 2: blink LED on/off with button	hub, LED, button	+ digital input, if/else control structures	Math/CS: logic
intro lab 3: use potentiometer to control an LED	hub, LED, potentiometer	+ analog sensing, analog mapping, (analog output control), variables, displaying incoming sensor values	Math/CS: value mapping; CVS
intro lab 4: make a colour sensor detect colours	hub, colour sensor	+ complex (3-dimensional) analog sensing and mapping	physics: additive colour sensing; light spectra
project: interactive baggage sort hallway mode: detect colour of baggage and give output signal	hub, RGB LED, colour sensor, piezo (optional)	applying previous concepts: conditionals, timing, value mapping	applying previous concepts in real-world complex system; additive colour mixing

Table 2: learning outcomes for the high school students scenario

analog sensing and output (mapping values, variables) in the third task. LED and potentiometer are used first because they allow for easy handling and control of the experiment (compared to the color sensor in the final project).

In the final project, there's more creative leeway to implement and customize the project. The project serves to apply the previous concepts in a real-world-like system. If students feel comfortable with the physical computing and programming, they can also use the piezo or other modules and get familiar e.g. with sound programming.

Engineering Scenario: Smart Home

The trials associated with students in engineering will open the possibility of performing tasks at a deeper level in the system than the other learning scenarios. It is not required, but possible, to hack both the IDE by adding new logical functions to it, and the firmware of the TALKOO boards to perform tasks not anticipated in the definition of the learning activity. Students in engineering are expected to have a deeper level of understanding of the technology.

General description of the activity

This scenario describes an activity for engineering students where they need to build a basic infrastructure for a smart home. They have different sensors and actuators to control events. The activity introduces the Arduino Visual Programming platform as a rapid prototype tool, provides learners with a

lab and project sessions	kit modules	programming / physical computing concepts	STEM and IxD concepts
intro lab 1: blink LED on/off with timer and button	hub, LED, button	timer (control) structures, digital output, digital input, if/else conditional structures	general: functional (visual) programming
intro lab 2: make a rotating pattern with LEDs	hub, LEDs	+ arrays, randomness	pattern/switch programming
intro lab 3: use light sensor to control a fan	hub, light sensor, relay	+ analog values, mapping values, programming relays	infrared sensing, circuiting with relays
engineering project: develop a smart home concept model	hub, piezo, light sensor, LEDs, buttons, temperature sensor, colour sensor, relay	apply previous concepts; + 3-dimensional analogue sensing and mapping (if colour sensor is used, e.g. for prototyping a vision-based alarm/authentication system)	applying and integrating concepts to build innovative complex system

Table 3: learning outcomes for the engineering students scenario

sequential learning sequence that leads up to an open ended project.

The main goal of this project is to introduce engineering students to diagrammatic programming, rapid prototyping skills, and notion of constraints as a design concept. The learning activity is divided into the introduction to the visual programming platform in a series of seven short instructed parts and then a more independent session where the groups of students develop their smart home concept.

The following building materials are provided: the PELARS Arduino kit, the IDE and additional project materials. As follows: a smart home model, tools, a mini fan and batteries. For programming, students can also modify the code behind the visual blocks.

Learning activities

There are three introductory lab sessions:

1. blink LED on/off with timer and button
2. make a rotating pattern with LEDs
3. use LDR to control a fan

Each of the introductory lab sessions comprises the same general steps described for interaction design above.

After the introductory lab sessions, students move on to the engineering project session. They are asked to develop and implement a smart home concept. Due to the limited size of PELARS kit modules and the workstation arrangement, they build their prototype in a box-like house model. Besides the kit hardware modules from the introductory lab sessions, they can also use other (more complex) sensors etc. from the kit, like the color sensor. In an iterative process of conceptualizing, building, programming, debugging, and documenting; students create their projects.

4 DESCRIPTION OF KIT “LAB WARE” MATERIAL

The PELARS kit is supposed to not only comprise hardware modules and IDE, but also complementary physical material to allow for specific STEM learning (see description of Task 4.4 in DOW). Based on the hardware kit and IDE development, previous work on user research in WP2 and integrated curricula in WP4, the above scenarios for learning activities were developed and refined involving especially educational project partners.

Further, Deliverable 2.2 “Recommendations on Learning Materials Research” identified several recommendations for complementary kit material. In summary, it was recommended:

- re-using and disassembling of (hardware) modules and material (tools),
- combination of standard reusable basic components to be extended with custom material, including recycling material, paper-based material etc.,
- material to fix/attach/combine hardware modules with other material, e.g. to move around or to build stable project setups,
- provide task-focused subsets of kits,
- provide basic parts for introductory tasks (without the need to craft) to explore basic STEM concepts for early rewarding experiences. These can be implemented in more depth, extended and customized later as part of personal-meaningful design projects.
- basic, flexible tools and material to allow teachers to customize kits with additional learning material for their individual educational purposes.

The material and resulting projects also need to be compatible to the hardware modules, they need to be small enough to fit on the workstations’ tables and within the range of the vision system.

According to the recommendations and proposed learning activities, complementary material for the PELARS kits has been designed and selected. The complementary kit material,

or “labware”, comprises a basic add-on set of tools and (removable) attachment material and three sets with parts for the specific learning contexts and their tasks (see figure 1). Tools and attachment material make the kit sustainable and reusable and customizable for individual making activities beyond the learning activities proposed below. The parts are the same for all three learning contexts. Task-specific kit sets vary for high school, interaction design and engineering education depending on the specific learning activities. Some task-specific parts (especially in the Interaction Design kit) are very raw and could

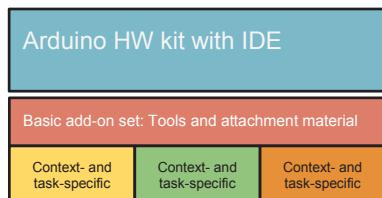


Figure 1: organisation of PELARS kit with kit subsets

be used beyond the proposed learning activities.

Throughout the course of PELARS, the trials and evaluation plan (D7.1) was redesigned for a more agile and iterative approach. It now contains initial phases of prototyping with teachers. The current design of “basic add-on set” with tools and attachment material and context- and task-specific sets allows the kit to adapt to the outcomes, e.g. changes in learning tasks and activities by only replacing task-specific parts.

Relation to Kit Learning Activities

Each learning unit or scenario comprises a set of learning modules that built up on each other: three to four introductory lab tasks and one to two “final” design projects (see above). The introductory tasks are very focused on individual concepts of programming and

lab and project sessions	kit hardware modules	add-on set material
intro lab 1: create simple sound	hub, piezo, button	none required
intro lab 2: make a melody	hub, piezo	none required
intro lab 3: make a random noise generator	hub, piezo	none required
design project 1: make a simple instrument (synth guitar)	hub, piezo, light sensor, 3 buttons	guitar neck, cardboard
design project 2: make a complex instrument (6-axis Theremin synth)	hub, piezo, accelerometer sensor, 2 buttons	cardboard, wearable computing material: needles, thread, Velcro, pieces of fabric

Table 4: description of the interaction design add-on set

physical computing. Due to the “plug-and-play” design of the hardware components and the IDE, they do not require additional physical material or tools to be completed.

The “projects” serve to apply the previously learned concepts, to involve related theoretical (STEM) concepts and to provide room for “making” (tinkering) activities with individual and collaborative problem solving. Therefore, the complementary kit material and tools support groups in individual designs of their projects that relate to real-world scenarios and can be personally meaningful to them. As building material, mostly cardboard and paper is used. This makes the material and shipping of kits low-cost, allows students to easily customize parts with common classroom tools and to combine with other crafting material (e.g., by easily cutting shapes into it, drilling holes for attaching hardware, painting, gluing other paper-based material on it).

All PELARS material and tool sets come in small boxes to be distributed one on each

workstation. Due to a potential diversity of individual projects (especially in interaction design education), the complementary building material may also be provided as a classroom set because not all students may need the same resources to customise their projects.

In the following, the proposed “basic add-on set” and the task-specific add-on sets are introduced for each educational context.

The basic add-on set

All kits contain a set of basic material and tools required mostly for the building of the design projects. These are flexible enough to be used for any other making activities with the hardware modules.

For mounting hardware modules on other material using the holes at the edges, attachment tools and material are provided:

- 2 mm (plastic) screws of different length and corresponding nuts
- a screw driver
- a small drill to make (additional) holes into cardboard
- coated wire as alternative to screws
- side cutting pliers to cut the wire (or cables in engineering projects)
- laser-cut cardboard part for the guitar neck; cardboard for the guitar body
- material for supporting wearables: needle, thread, safety pins, Velcro, pieces of fabric.



Figure 2: tools and attachment material (see descriptions in picture)

For customizing project parts with cardboard, paper or fabric, additional tools and material are provided (see figure 3):

- a roll of tape and glue to fix cardboard
- scissors to be used with paper, fabric etc.
- a cutter to cut out custom cardboard pieces (e.g., in interaction design projects) or to customise the cardboard models (e.g., in engineering projects)

Additionally, the context- and task-specific material sets are provided to support the specific learning activities (final projects) of an educational context. These are presented in the following sections.

Context- and task-specific material sets

Interaction Design material set

In summary, the interaction design set (see figure 3) provides additionally to the basic add-on set described above (see table 4):



Figure 3: material in Interaction Design set: guitar neck template, plain cardboard, fabric, strap, Velcro, sewing reel and needle, safety pins

For the synth guitar project, a guitar neck has been designed with holes to attach the hardware modules (figure 5). It is manufactured out of corrugated paperboard (4mm) using a laser cutter. The digital templates could be provided to the design students to modify and cut out with their own laser cutter.

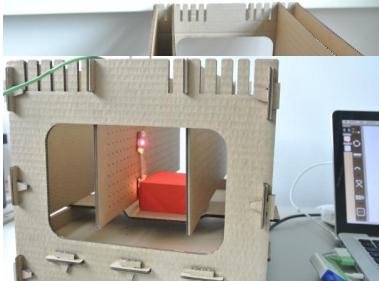
For the synth Theremin project, students have more freedom to design a musical instrument. This is reflected in the material provided. At the heart of the project is the accelerometer module. An accelerometer is used to measure acceleration and thus affords to be moved to give meaningful output. The hardware module design with holes at the sides makes them usable for wearable projects and allows to combine parts with textile material by sewing into the holes. Therefore, textile material is provided to offer students this option for their design project: a piece of fabric, strap, Velcro, safety pins, sewing reel and needle. But also the



Figure 4: guitar neck with hardware modules attached to holes

lab and project sessions	kit hardware modules	add-on set material
intro lab 1: blink LED on/off with timer	hub, LED	none
intro lab 2: blink LED on/off with button	hub, LED, button	none
intro lab 3: use potentiometer to control an LED	hub, LED, potentiometer	none
intro lab 4: make a colour sensor detect colours	hub, colour sensor	paper of different colour
project: interactive baggage sort hallway mode: detect colour of baggage and give output signal	hub, RGB LED, colour sensor	box template for hallway model, coloured paper templates for making baggage models

Table 5: description of the high school add-on set



models (here)
Figure 6: hallway model in action with hardware module attached to inner walls and red baggage model on conveyor belt

plain cardboard of the Interaction Design set can be used to build the Theremin synth project, e.g. for building instruments that can be shaken and moved (like e.g. a tambourine). In general, interaction design students are

encouraged to complement with tools and material from their lab facilities like laser cutters.

High School material set

In summary, the high school set (additionally to the basic add-on set described above – see table 5) provides (see figure 5):

- parts for the hallway model with conveyor belt
- coloured paper for creating baggage paper models

The hallway box was specifically designed for the high school project (figure 5, figure 7). It is

house is very flexible and supposed to be customised by the students. The house consist of two storeys and a roof. The storeys and the roof are stackable so that students can easily modify the model to their project design. A division can be inserted and moved to divide a storey into two rooms. The outer walls of the house do not contain holes for doors or windows and no holes for attaching hardware modules yet. Students are supposed to cut and drill them by themselves according to their project design. Since they may want to use a light sensor, pre-given holes may affect the

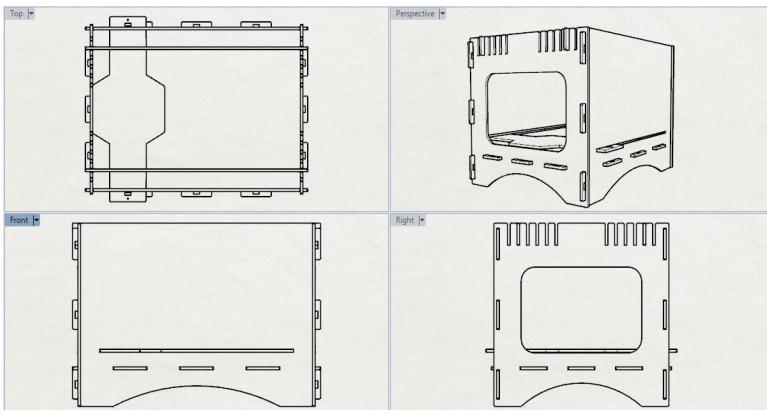


Figure 7: CAD drawing of the hallway box

made of stable cardboard and has two extra walls with holes for attaching the hardware components. These walls can be moved to adjust the distance to the colour sensor (figure 6).

Engineering Education material set

In summary, the engineering set (see figure 8) provides additionally to the basic add-on:

- fan (motor) and power source (batteries and battery holder, cables);
- cardboard “house” box template (to represent home model)

The smart home was specifically designed for the engineering project. It is made of stable 4mm corrugated cardboard. The design of the

light conditions inside the house.



Figure 8: engineering set: smart home model (with stackable stories, roof and room division) and mini fan with power source (battery holder with batteries) and cables

lab and project sessions	kit hardware modules	add-on set material
intro lab 1: blink LED on/off with timer and button	hub, LED, button	none
intro lab 2: make a rotating pattern with LEDs	hub, LEDs	none
intro lab 3: use light sensor to control a fan	hub, light sensor, relay	mini fan, batteries, battery holder, connection cables
engineering project: develop a smart home concept model	hub, piezo, light sensor, LEDs, buttons, temperature sensor, colour sensor, relay	smart home model, mini fan, batteries, battery holder, connection cables

Table 6: description of the engineering add-on set

5 TALKOO KIT'S INNOVATION WITHIN EDUCATION

TALKOO has been designed to allow users start building electronic devices without having to build circuits neither on breadboards nor on prototyping boards, without the use of jumper wires, and without having to write complex lines of code. This doesn't mean that it is not possible to approach the resolution of a problem, with TALKOO, using the above-mentioned more traditional techniques; it means that we have designed and implemented a higher level of abstraction around educational electronics that allow yet another way of confronting the realization of projects.

Hot-plugging Mechanism

From a technical standpoint, each one of the TALKOO modules is nothing but a microcontroller board with a single sensor or actuator connected to it and making use of a bus communication protocol to handle acquiring information and sending it

to/receiving it from a computer. From a user point of view, what we have developed is a system where sensors and actuators can be hot-plugged (that is without having to restart the system) to the computer.

Visual Programming

Each one of the modules will have a software equivalent, called blocks. The blocks will be related to each other by drawing connections between each other. For example, a button block will have an output that could be connected to any device having inputs. There are blocks like the one for the LED that have inputs. By drawing a line between the output from the button block and the input of the LED block, we will establish a relationship between the physical modules represented by the blocks. The information from the first one will be sent to the input from the second one. All of these interactions will be described on the IDE

(Integrated Development Environment) created for the project. This IDE allows to both change the relationships between the different modules connected by making what we call the taxonomy of the project, but also reprogramming the firmware of each one of the different modules by means of an external hardware programmer.

Mix of Embedded and PC Computation

In order to take advantage of having the TALKOO kit connected to a computer, we have implemented a mechanism for introducing more complex computational modules that will operate inside the computer running the IDE. There are a series of blocks in the development environment that can perform all sorts of logical functions.

It is possible for the advanced user to create his/her own blocks in order to enhance the kind of operations to perform on the system. At the time of writing, this will require hacking into the source of the IDE, but since it is written in JavaScript, it is not a big issue, given that the person doing it knows how to program that language.

There is also a special mechanism to patch whatever information comes from the hardware modules directly to a third-party program. We have implemented a way for other programs to connect to the system, what allows developers to use whatever software development tool they are comfortable with to work with the information coming from the different sensors and actuators and bypass the given IDE tools.

Technical Solution

This section describes each one of the parts of the technical solution in depth, from the hardware prototypes to the software ones. We make more emphasis in the hardware developed as this is the final form it will have. At the time of writing the prototypes have been tested for proper functionality and if there is any work left to be done it is at the software level, and will be performed as we try the kits

with different users over the forthcoming months.

Hardware Prototypes

In total 13 module types have been designed, only a few of them had to go through different design iterations, but once we had established a framework for how to design new modules, the exercise of creating new designs was almost free of big failures.

The main characteristics of the TALKOO kit are:

- modules' size is a multiple of 25mm in both x and y axis
- modules have no components on the B side, what allows soldering modules on top of a larger container board
- modules are connected with a 5-wires cable carrying: power, ground, an I₂C bus, and a wake-up pin (which functionality was not implemented at the time of writing)
- they have an 8-bit micro-controller operating at 8MHz and 3V3, 32K of flash, 512b of ROM and 4K of RAM; the micro-controller has an internal 10b ADC, SPI, I₂C, and three oscillators that can be used to perform PWM output on 6 of the pins
- in total there are 12 modules
 - rotary encoder
 - potentiometer
 - motor controller
 - relay
 - LED
 - RGB LED
 - 6 axis MEMS
 - piezo
 - colour sensor
 - light sensor (photodiode, not LDR)
 - temperature
 - button
- besides the modules, there is a so-called hub, which makes a simple I₂C to USB conversion to allow the

modules communicate with the IDE in a simple way

between 0 and 100 depending on the rotation versus a certain reference angle.

Module: Rotary Encoder

The rotary encoder is a digital input device that can detect both angular rotation and the press of the knob in a vertical fashion. The device will send through the bus a number indicating the amount of rotations in a certain direction. When rotated clockwise, the rotation value increases while, when rotated counter clockwise, the rotation value decreases.

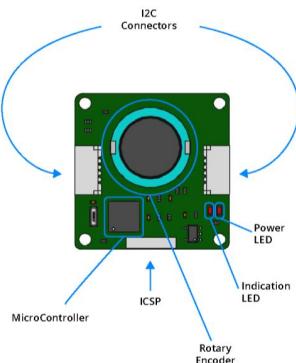


Figure 10: illustration of rotary encoder, top view and description of parts

Thus, the direction of rotation is indicated by the sign of the increment. This means as well that the rotation value could be negative.

The second interaction mode, besides rotating the shaft of the encoder, it is to press it downwards. The device has a switch that will be detected when pressed. The firmware has been implemented in a way so that when the shaft is pressed, it will reset the counter to zero, no matter what the rotation value is at that point.

The measurement of this sensor is done entirely through digital signals, it should be possible to rewrite the firmware for it to support a similar behaviour as the potentiometer, e.g. sending a fixed set of values

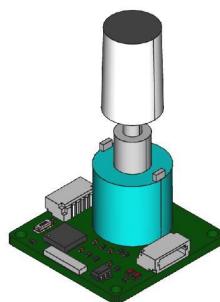


Figure 11: isometric illustration of rotary encoder



Figure 9: photograph of the rotary encoder module, mounted by hand

Module: Potentiometer

The potentiometer is an analog input device that detects the angular rotation by means of reading the change in the voltage on a variable resistor connected to an ADC enable input on the module's microcontroller.

That value is measured and sent via the bus when requested by the hub. The potentiometer we found for developing our modules, doesn't

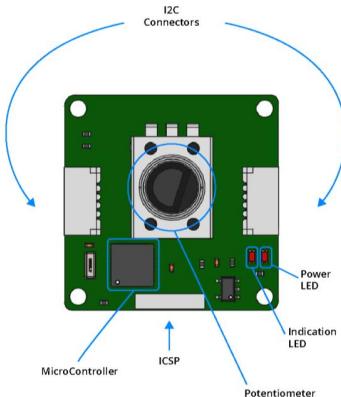


Figure 12: illustration of potentiometer, top view and description of parts

come with a shaft and therefore we have created a 3D model of a shaft that is part of the TALKOO kit.

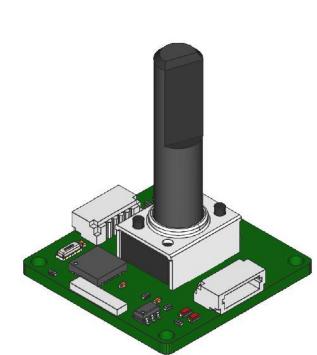


Figure 13: isometric illustration of potentiometer

The measurement of this sensor is done through analog signals, what makes it less accurate than the one from a rotary encoder, however, it is very simple to use and makes the possibility of hacking the functionality of the module very easy for the advanced users.



Figure 14: photograph of potentiometer, mounted by hand

Module: Motor Controller

The motor controller is an output device including a motor driver chip (also known as H-bridge) that allows controlling:

- 4 DC motors with variable speeds, one direction of rotation, or
- 2 DC motors with variable speeds and variable direction of rotation, or
- 1 stepper motor

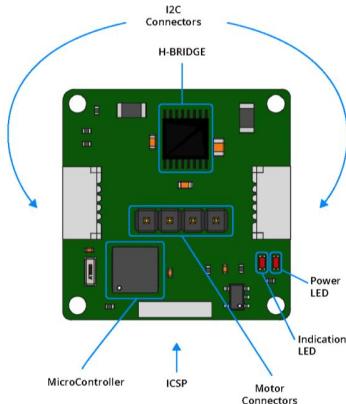


Figure 15: illustration of motor controller, top view and description of parts

The motor controller takes signals sent from the IDE to command the outputs in different ways. The main limitation of the current design of this module is that the motors have to be

powered at 5 volts, since the power for them is taken from the standard communication cable and not from an external source. This design

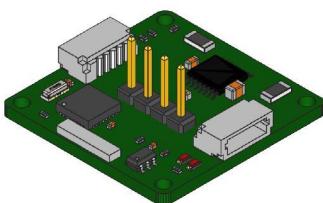


Figure 16: isometric illustration of motor controller

might be redone in future revisions of the kit to accommodate other sources of power for motors.



Figure 17: photograph of motor controller, mounted by hand

Module: Relay

The relay module is an output device including a physical relay that can switch up to 250VAC/5Amp with a low voltage input. The relay exposes three connectors: the common one, one that is normally connected and that is normally opened. In this way, it is possible to let power go through by default or vice-versa, giving maximum versatility to the module.

This module can be used to control a whole range of home appliances by simply connecting the power cable to it as a sort of software controlled power switch.

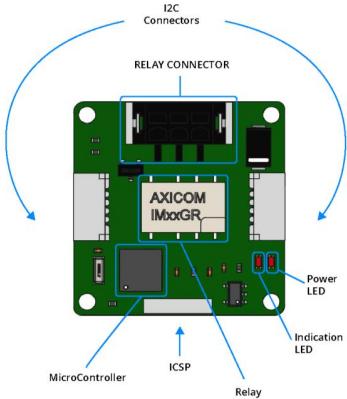


Figure 18: illustration of relay, top view and description of parts

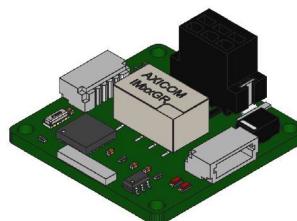


Figure 19: isometric illustration of relay

Module: LED

The LED module is the simplest output device possible, it allows controlling a single one-color LED. The module gets values through the bus to control not just whether the LED is on or off, but also a specific level of light on the LED.

We anticipate that this module will be one of the most used when designing simple systems and we think that kits should contain multiple LEDs.



Figure 20: photograph of relay, mounted by hand



Figure 23: photograph of LED, mounted by hand

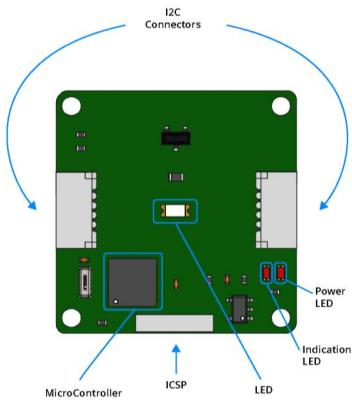


Figure 21: illustration of LED, top view and description of parts

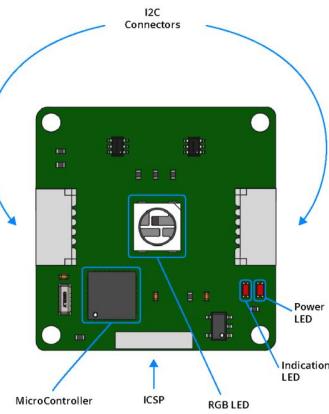


Figure 24: illustration of RGB LED, top view and description of parts

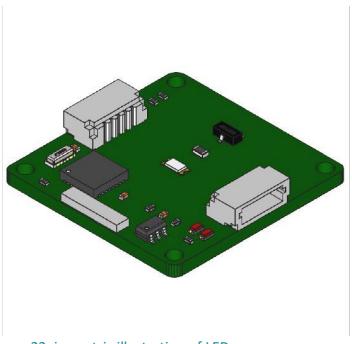


Figure 22: isometric illustration of LED

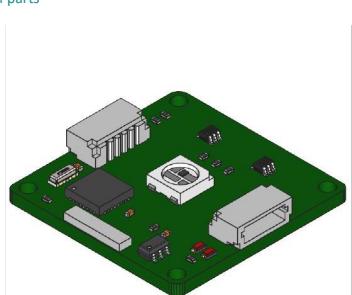


Figure 25: isometric illustration of RGB LED

Module: RGB LED

The RGB LED module is an output device that allows representing any kind of colour with a single module. It gets information from the IDE in terms of percentages of colour red, green and blue and uses those to trigger different pulse modulated signals on the inputs of the on-board RGB LED.



Figure 26: photograph of RGB LED, mounted by hand

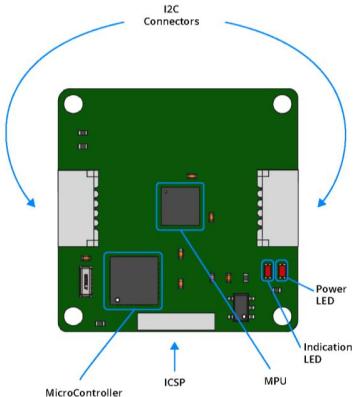


Figure 27: illustration of 6 axis MEMS, top view and description of parts

Module: 6 Axis MEMS

The 6 axis MEMS is an input device with a Micro-Electro-Mechanical System (MEMS) that acts as a combined accelerometer and gyroscope sensor. This module can be addressed with a series of commands that will indicate which kind of information we want to read from it every time we send a request from the IDE.

As a sensor it can be used for reading information about inclination angle and rotation speed in three directions of space. The information of such a sensor can be used to compute more complex information like step counting, detection of a free fall situation, accurate relative positioning, etc.

The information gathering is done via a digital communication protocol between the module's microcontroller and the MEMS sensor. The micro-controller will store it and serve it when requested by the IDE.

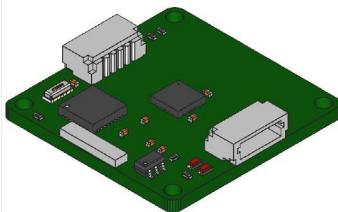


Figure 28: isometric illustration of 6 axis MEMS



Figure 29: photograph of 6 axis MEMS, mounted by hand

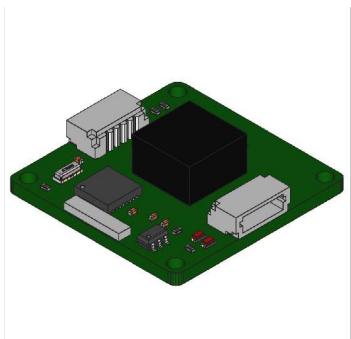


Figure 31: isometric illustration of piezo

Module: Piezo

The piezo module is an output device aimed at playing tones when instructed from the IDE. It is possible to control the frequency at which the piezo will oscillate what makes it useful for giving auditory indications, playing small melodies, and even as a sort of mechanical vibrator (when operated at very low frequencies).

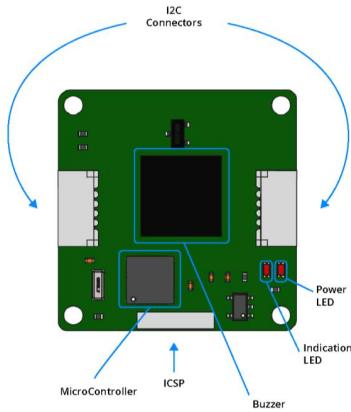


Figure 30: illustration of piezo, top view and description of parts



Figure 32: photograph of piezo, mounted by hand

Module: Color Sensor

The colour sensor is a module with a white light LED and a colour sensor chip. This module can capture the intensity of light separating the light in three different segments of the spectrum: red, green and blue. It will give the percent value of each colour that could in turn be used to e.g. light up an RGB LED with the same colour captured by the sensor.

The module has, besides the white LED, a TCS3472 colour sensor that provides a digital return of red, green, blue (RGB), and clear light sensing values. The sensor filters away the infrared light (IR), giving a very accurate measuring value via a digital protocol.

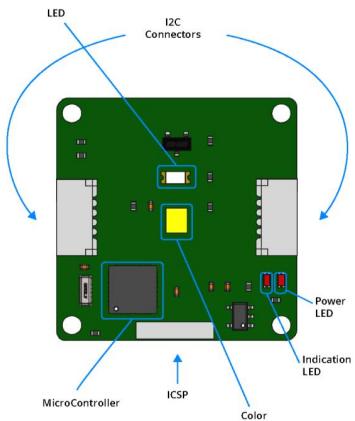


Figure 33: illustration of colour sensor, top view and description of parts



Figure 35: photograph of colour sensor, mounted by hand

Module: Light Sensor

The light sensor module is an input device that can read the intensity of light using the microcontroller's internal ADC peripheral. Unlike other models of light sensors available, we have decided to use a photodiode instead of an LDR due to the higher accuracy in the measurement of light and the ease of calibration of the sensor.

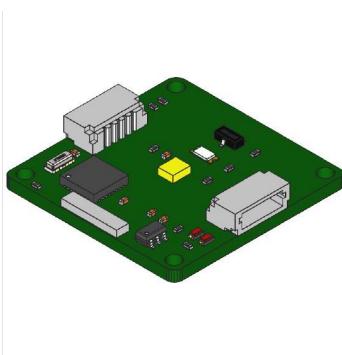


Figure 34: isometric illustration of color sensor

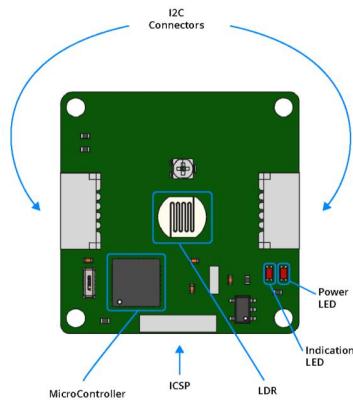


Figure 36: illustration of light sensor, top view and description of parts

The chosen active device is the TSL13T a highly integrated light-to-voltage optical sensor, combining a photodiode and a transimpedance amplifier on a single integrated circuit.

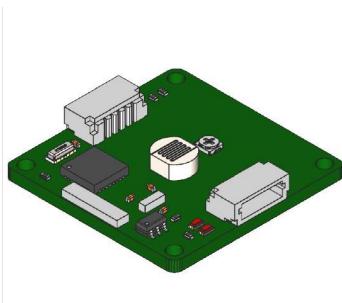


Figure 37: isometric illustration of light sensor

The module is, however, labelled as LDR since it is the way many users currently understand and name light sensors. This will be renamed in the future to accommodate the actual part used to sense light.



Figure 38: photograph of light sensor, mounted by hand

Module: Temperature Sensor

The temperature sensor module is an input device that can read the current temperature on the air surrounding the sensor. The sensor used is the TMP102, a highly sensitive (0.5 degrees of accuracy without calibration) part that communicates via I2C to the main processor on the board.

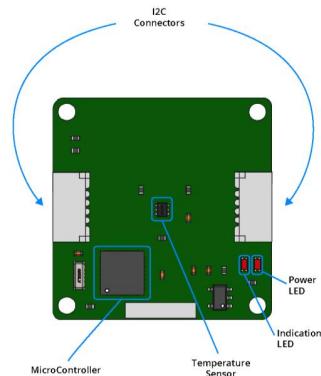


Figure 39: illustration of temperature sensor, top view and description of parts

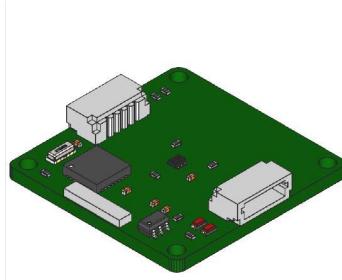


Figure 40: isometric illustration of temperature sensor



Figure 41: photograph of light sensor, mounted by hand

Module: Button

The button module is an input device that can be used to detect and classify the user interaction with a simple tactile switch. At the time of writing, the default button firmware did only allow simple press-release interactions, but it offers the possibility of implementing other detection patterns like double click, continuous press or even use different debounce strategies.

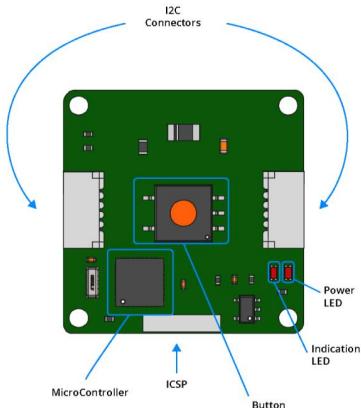


Figure 42: illustration of button, top view and description of parts

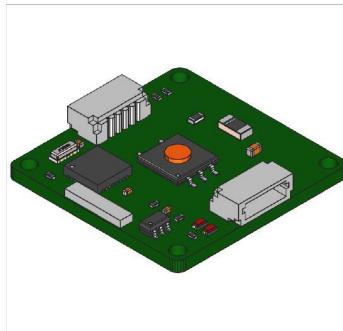


Figure 43: isometric illustration of button



Figure 44: photograph of button, mounted by hand

Module: Hub

The hub is a special type of module used to get the chain of input/output modules to exchange information with the computer. It could be described as a I2C to USB converter, but with enhanced functionality.

As mentioned earlier, one of the main technical innovations behind the TALKOO kit is the hot-plugging communication protocol. The hub plays a major role in this case, as it commands the address assignation and reports back to the computer about how the device ID corresponds to a certain BUS ID. This is what allows to uniquely identify each one of the modules connected to the system.

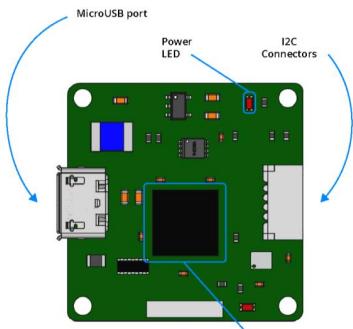


Figure 45: Illustration of hub, top view and description of parts

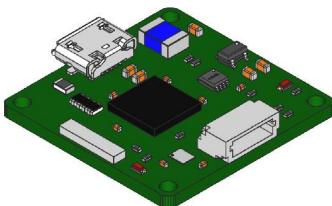


Figure 46: Isometric illustration of hub

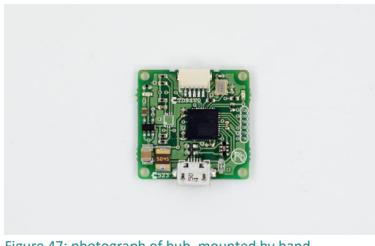


Figure 47: photograph of hub, mounted by hand

Software Development

The TALKOO kit comes with a visual programming interface built as a tool to the standard Arduino IDE from the 1.6.X family. As explained in previous deliverables (D4.1, D5.1) the Arduino IDE is a Java application. The IDE can be extended with different tools and off-the-shelf Java libraries. As from Java's runtime environment 1.8 it is possible to render HTML5 code on top of a special Java canvas taken from the Java FX library.

This means that, even if the TALKOO Arduino software tool runs on a Java IDE, it has been completely designed and coded in JavaScript as a way to be forwards compatible with forthcoming versions of the Arduino IDE that will be HTML5 based. In turn this allows to e.g. run the IDE inside a web browser.

The visual programming environment has a toolbar, a canvas, a properties window and a play button. Each one of those elements are described in a later section.

The main thing the IDE is used for is to draw the interaction between modules and between those and logical blocks. Therefore, the toolbar allows selecting between different types of blocks: some of them represent the physical modules, while some others represent logical functions. The toolbar has three different tabs: physical modules, simple logical functions and games.

The games tab is used for including complex blocks that have been specifically developed for the demo session that PELARS had at the Ars Electronica Festival 2016. In this specific setup it featured the games: Pong and Simon Says, which can be easily be implemented by using TALKOO modules.

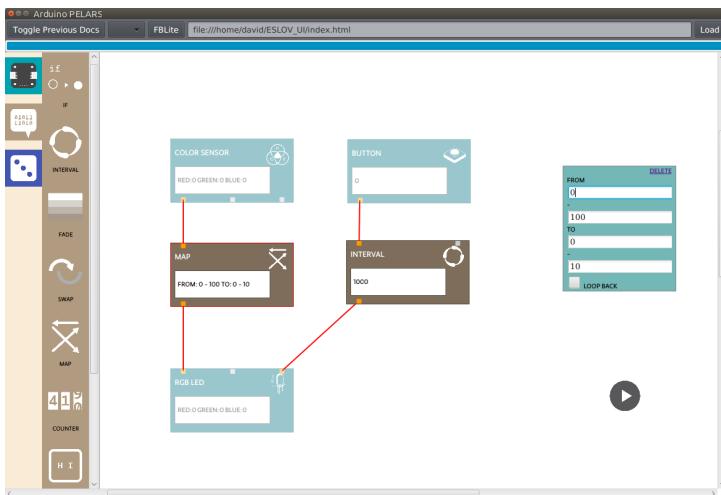


Figure 48: screenshot of the TALKOO tool for the Arduino IDE

The Physical Blocks

Table 7 shows the graphical representation of each one of the physical modules.

Module	Type	Icon
rotary encoder	input	
potentiometer	input	
motor controller	output	

relay	output	
LED	output	
RGB LED	output	
6 axis MEMS	input	

piezo	output		map	given a value at the inlet and a range of values at the parameter dialog, will map the input into a different range	
colour sensor	input		swap	will change between values given as parameter, it is possible to introduce a relatively long array of values	
light sensor	input		trigger	will send whatever is given as parameter when the inlet is triggered	
temperature	input		interval	when the inlet is triggered, it starts sending events to its outlet with a period given as parameter	
button	input		fade	used to increase or decrease a counter between two values at a certain pace	

Table 7: graphical representation of the physical modules in the VPL.

The Logical Blocks

The initial amount of logical blocks developed at the time of writing is 11, being 2 of those games that will be explained in the next subsection. The others are shown in table 8.

Block	Description	Icon
if-then-else	takes one input and given a comparison that can be written in the parameters dialog, will send an output through its left or right outlet	
note	given an event on an inlet, sends a note value to an outlet, ideally is used to play sound when connected to the piezo module	

random	generates a random number	
--------	---------------------------	---

Table 8: graphical representation of the logical modules in the VPL

The Games

For the purpose of the demo that PELARS performed at Ars Electronica Festival 2016, we developed two games that used physical modules as ways to interact with the system and a single block to contain all of the game logics.

This was an attempt of testing:

- physical affordances of the bare modules, without having casted them into anything

- the complexity of creating new blocks for the visual programming language, a need for the more advanced user cases to be performed with engineering students later in the project

At the time of writing there are still no results to the process of testing the modules, however it was possible to create the game logics as blocks in a relatively short amount of time, what proves the point that the system has been designed in a way that allows for software improvements. It still has to be documented how users can do that by themselves, though.

The two games created for this demo were Simon Says and Pong (figure 49 shows, however, a third game: Theremin, which was developed for the conceptual prototyping phase). Both are classic games with clearly understood game logics by most adults. The idea behind the experiment is to let kids mount the physical interface to the game (with some adult supervision), then drag the modules into the visual IDE, connect the parts, and start playing the game.

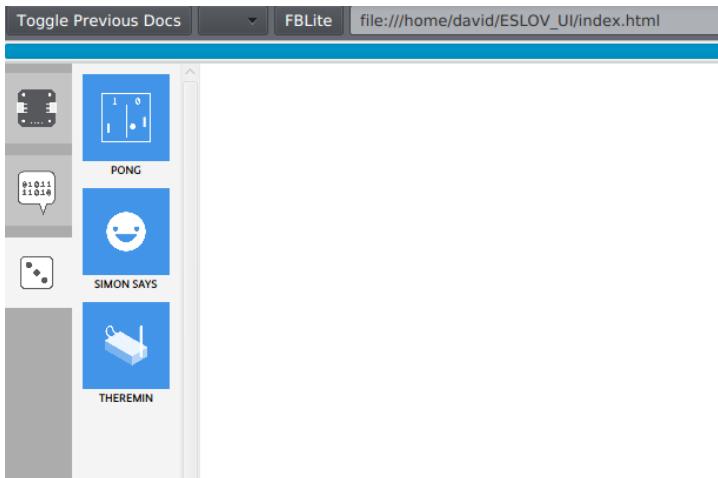


Figure 49: TALKOO gaming menu as created for Ars Electronica 2015

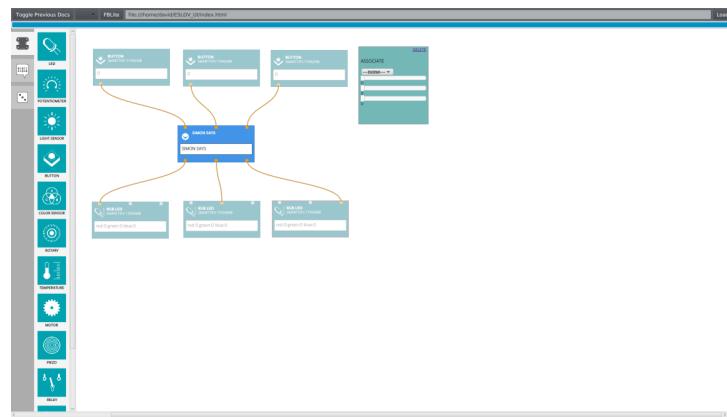


Figure 51: Simon says as made for Ars Electronica 2015

The games have been incorporated into the IDE by placing them under their own tab. We think there will be room for more additions like this one and even for the ones made by the participants in the trials, therefore it is important to reserve a dedicated location in the IDE for them.

Simon Says

Simon Says (figure 49) is a classic game where players compete against a machine. The machine will play a sequence of sounds and lights (with matching tones for each light colour) and the player is supposed to reproduce it. An error in introducing the sequence or a timeout will make the player lose the game.

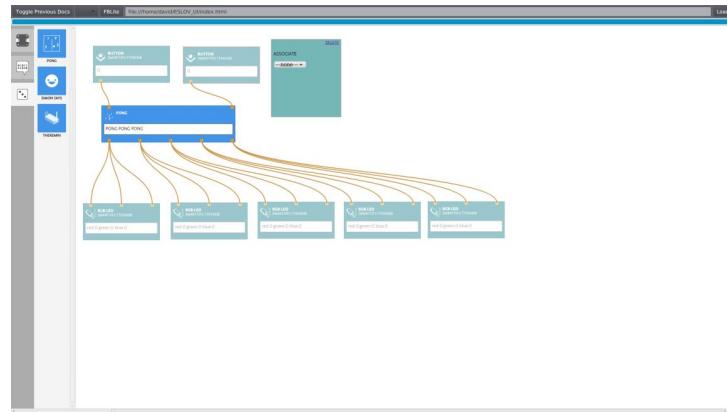


Figure 50: Pong as made for Ars Electronica 2015

This is very easy to create using the TALKOO kit. You need just a few buttons, a few RGB LEDs and a piezo module.

Pong

Pong (figure 50) is one of the first two player computer games ever made. In this game, players send a virtual ball across a table-tennis board on the screen and try to hit it with a virtual paddle. The physical computing version of Pong is one-dimensional, it is made of a row of LEDs and players will have to "hit the ball" by pressing a button when the light reaches their end of the LED row. Summarizing, Pong is made of a series of RGB LEDs and two buttons.

Backend vs UI

Regarding the visual programming environment it is important to note that it has been developed as a two-layered system. There is a backend that is responsible for keeping the communication towards the modules alive, channelling the data between modules, etc. And there is a frontend (also called User Interface or UI among the project developers) that represents the modules, runs the animations, and has room to incorporate data visualizations coming from other workpackages (WP2) into the IDE.

This separation was required for development reasons, to ensure the portability of the code to other systems. This will allow, in the future, to consider entirely different ways to represent the blocks and to provide a foundation for porting the system from PCs to touch interfaces with a different usage pattern without running the risk of entangled code-spaces. At the same time allows for separating the physical processes from the graphical ones, in other words, the communication between modules must have a higher priority -as it might be operating devices with motors, or switching high voltage electrical currents- than representing blocks on the computer screen. Separating both systems helps assigning higher priority to those processes that need it most.

Backend: Serial Communication

The communication between the backend and the hub is made through a virtual serial port. This is a very specific solution that has required a different strategy for the Arduino IDE tool development than for running the visual IDE separately inside a browser. The communication has been tuned to push down the latency per module to be as low as 20ms. This means that the more modules are connected to the system, the slower it will become.

Therefore, at the current state of development, even if it is possible to connect up to 127 simultaneous modules to the backend, the latency could become so big that it would make the installation unusable from a UI point of view. The bottleneck in this case is not the serial communication, that has been optimized, but the inter-module communication. This is not an issue for PELARS, but it is something that Arduino is already looking for a later commercial exploitation of the research results. Currently it is possible to add up to 30 modules to the chain without affecting the communication too much. This is explained further later in the firmware development section of the text.

Backend: Websocket Tool

Since there is a separation between backend (communication) and frontend (visualization), it is possible to use the same technique to allow other programs connecting to the IDE. Therefore we have added a feature to the backend in the form of a websocket that other programs can connect to, so that it is possible to implement mainly external visualization systems.

Firmware Development

The PELARS' communication protocol is an I2C protocol running in multi-master mode with a transport layer encoded using Concise Binary Object Representation (CBOR¹) codification. The I2C bit rate for this protocol is set at 0,1 Mbit/s, this cannot be improved much

¹ CBOR stands for Concise Binary Object Representation and it is a way to encode information in a binary format that allows direct parsing of information upon arrival from a

sending system into a receiving one. It is optimal for encoding information sent between devices with small computing power (like the TALKOO

further at the time of writing due to the type of processor chosen for the PELARS modules. The networking model implemented for assigning addresses to the modules is modelled as if it was a sort of “Dynamic Host Configuration Protocol” (DHCP), this is what allows the system to hot-plug modules without the need for a reset.

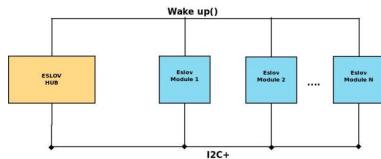


Figure 52: diagram depicting the communication lines between hub and modules

During module detection, the protocol also assigns a unique ID, and in the current version, it is possible to connect up to 30 different TALKOO modules at the same time. This concept is what makes the PELARS protocol different from a standard I2C one. Therefore we decided to give it the name I2C+, as it is somehow better than a completely standard I2C protocol.

The hot-plugging and UID assignment mechanism has been implemented in the hub. The IDs range 2 to 31, keeping the hub the ID number 1. A standard I2C protocol requires just 4 wires for its implementation, being two of them power and ground. Ours has a fifth wire, called wakeup, add for future compatibility of the protocol with future revisions that would allow the different modules to initiate the communication by forcing an interrupt on the port when grounding the wakeup pin.

Packet structure

This section looks at how information is structured when sent between the hub and the different modules. As the modules have

modules) since it allows for a smaller codebase and memory footprint for de/encoding operations.

different capabilities, the kind of data sent

ID	Type of Module	Data Payload
----	----------------	--------------

Figure 53: I2C+ package structure

differs in each case, even if the basic structure remains the same. The packet structure is defined in a static way with a size of up to 30 bytes of information. To optimize it, The PELARS packet is binary codified with CBOR, a binary data format based in and 100% compatible with JSON data objects. The data structure is divided into head (coloured in grey) with fixed size of 2 Bytes and PAYLOAD (orange) with variable size up to 28 Bytes.

Header

The header has two mandatory fields:

- ID: Unique module identification between 1 until 31(HUB gets always ID 1).
- ToM: Information about the type of module, currently a number from 0...12 modules (HUB always 0).

Modules' Payload Configuration

The payload has variable size and contains different fields depending on the module involved in the communication. The following subsections describe the specific payload for each one of the PELARS modules.

PELARS Button Payload

The button module is a sensor module with one field in the payload with a size of 1 Byte. It has been designed with two states.

ID	Type of Module	Button status
----	----------------	---------------

Figure 54: payload for the button module

- **Button status (1 Byte):** An integer value which contains the status of the

button, two possible values are possible: 0 is unpressed and 1 is pressed.

PELARS LED Payload

The LED module is an actuator module with one field in the payload with a size of 1 Byte.

ID	Type of Module	LED status
----	----------------	------------

Figure 55: payload for the LED module

- **LED status (1 byte):** An integer value which contains the status of the LED, two possible values are possible: 0 is for when the LED should be turned off and 1 is for when the LED should be on.

PELARS LDR Payload

The LDR module is a sensor module with one field in the payload of up to 3 Bytes.

ID	Type of Module	LDR status
----	----------------	------------

Figure 56: payload for the light detection module

- **LDR status (up to 3 bytes):** A positive integer value (0...1023) which contains the actual value of the light sensor.

PELARS PIEZO Payload

The piezo module is an actuator module with two fields in the payload of up to 6 Bytes of size.

ID	Type of Module	Note Frequency	Note Duration
----	----------------	----------------	---------------

Figure 57: payload for the piezo module

- **Note Frequency (up to 3 bytes):** A positive integer value (0 .. 16.000khz), it contains the actual value of the frequency for the note to be played.
- **Note Duration (up to 2 bytes):** A positive integer value (0, 2, 4, 8, 16,

32) which contains the note's duration. 1 is the longest duration.

PELARS RGB Payload

The RGB LED module is an actuator module with three fields in the payload of up to 6 Bytes of size.

ID	Type of Module	Red LED Value	Green LED value	Blue LED value
----	----------------	---------------	-----------------	----------------

Figure 58: payload for the RGB LED module

- **Red LED value (up to 2 bytes):** A positive integer value (0...255) which contains the value to write to the red LED using Pulse Width Modulation (PWM).
- **Green LED value (up to 2 bytes):** A positive integer value (0...255) which contains the value to write to the green LED using PWM.
- **Blue LED value (up to 2 bytes):** A positive integer value (0...255) which contains the value to write to the blue LED using PWM.

PELARS Color Detector Payload

The colour detector module is a colour sensor with three fields and up to 9 Bytes of size.

ID	Type of Module	Red LED Value	Green LED value	Blue LED value
----	----------------	---------------	-----------------	----------------

Figure 59: payload for the color sensor

- **Red Sensor value (up to 3 bytes):** A positive integer value (0...1023) which contains the value of the red sensor.
- **Green Sensor value (up to 3 bytes):** A positive integer value (0...1023) which contains the value of the green sensor.
- **Blue Sensor value (up to 3 bytes):** A positive integer value (0...1023) which contains the value of the blue sensor.

A special configuration package can be sent from the HUB to the module in order to activate the flashlight of the colour detector.

ID	Type of Module	Flash Lighting
----	----------------	----------------

Figure 60: payload for the flash light activation on the colour sensor module

- **Flash Lighting (1 byte):** An integer value which contains the action to activate or deactivate the flash lighting of the colour detector module. 0 is to turn the LED off and 1 is to turn it on.

PELARS Temperature Payload

The temperature module is a sensor module with one field in the payload of up to 3 Bytes.

ID	Type of Module	Temperature value
----	----------------	-------------------

Figure 61: payload for the temperature module

- **Temperature value (up to 3 Bytes):** A positive integer value (0 .. 1023) which contains the actual temperature on the sensor, an additional formula is applied in the backend to transform that information into Celsius degrees.

PELARS Motor Control Payload

The motor control module is an actuator for control up to 2 DC motors. The payload has 4 fields to control the PWN signals on the pins of the motor driver.

ID	Type of Module	MOTOR 0 PIN A	MOTOR 0 PIN B	MOTOR 1 PIN A	MOTOR 1 PIN B
----	----------------	---------------	---------------	---------------	---------------

Figure 62: payload for the motor control module

- **Motor 0 Pin A (up to 2 bytes):** A positive integer value (0 .. 255) which contains the value to write into the PIN A of the Motor 0 using PWM.
- **Motor 0 Pin B (up to 2 bytes):** A positive integer value (0 .. 255) which

contains the value to write into the PIN B of the Motor 0 using PWM.

- **Motor 1 Pin A (up to 2 bytes):** A positive integer value (0 .. 255) which contains the value to write into the PIN A of the Motor 1 using PWM.
- **Motor 1 Pin B (up to 2 bytes):** A positive integer value (0 .. 255) which contains the value to write into the PIN B of the Motor 1 using PWM.

PELARS Potentiometer Payload

The potentiometer module is a sensor module with one field of up to 3 Bytes.

ID	Type of Module	Potentiometer position
----	----------------	------------------------

Figure 63: payload for the potentiometer module

- **Potentiometer position (up to 3 Bytes):** A positive integer value (0...1023) which contains the actual analog reading of the potentiometer.

PELARS Relay Payload

The relay module is a actuator with one field in the payload with a size of 1 Byte.

ID	Type of Module	Relay position
----	----------------	----------------

Figure 64: payload for the relay module

- **Relay status (1 byte):** An integer value which contains the status of the Relay, two possible values are possible: 0 is to keep the relay in the default position and 1 is to activate.

PELARS Rotary Encoder Payload

The rotary encoder is a sensor module with a payload of two fields, that added up could have a total length of up to 4 bytes.

ID	Type of Module	Rotary position	Button Status
----	----------------	-----------------	---------------

Figure 65: payload for the rotary encoder module

- **Rotary position (up to 3 Bytes):** An integer value which contains the number of steps per direction of the rotary encoder.
- **Button status (1 Byte):** An integer value which contains the status of the button, two possible values are possible: 0 is for when the button is unpressed and 1 for when it is pressed.

PELARS Accelerometer Payload

The accelerometer and gyroscope module is a sensor. The payload has 6 fields with gyroscope and accelerometer information.

ID	Type of Module	Accelerometer X	Accelerometer Y	Accelerometer Z	Gyro X	Gyro X	Gyro Z
----	----------------	-----------------	-----------------	-----------------	--------	--------	--------

Figure 66: payload for the accelerometer and gyroscope module

- **Accelerometer X (up to 3 bytes):** An integer value which contains the digital output for X-axis Accelerometer in $\pm 16g$ scale.
- **Accelerometer Y (up to 3 bytes):** An integer value which contains the digital output for Y-axis Accelerometer in $\pm 16g$ scale.
- **Accelerometer Z (up to 3 bytes):** An integer value which contains the digital output for Z-axis Accelerometer in $\pm 16g$ scale.
- **Gyroscope X (up to 3 bytes):** An integer value which contains the digital output for rate angular sensor X-axis Gyroscope.
- **Gyroscope Y (up to 3 bytes):** An integer value which contains the digital output for rate angular sensor Y-axis Gyroscope.
- **Gyroscope Z (up to 3 bytes):** An integer value which contains the digital output for rate angular sensor Z-axis Gyroscope.

Visualization Tools

Project partner CIID is designing and developing a visualization tool that will take in the data streams created from the PELARS table and students, pass these through established Learning Analytics pathways, and finally result in a dashboard display that will contain modules of these processed data streams. This work builds on top of D4.1 where the initial ideas for the information visualization tool were presented. The Arduino PELARS kit provides us with data about the students' use of the hardware modules along with manipulations in the visual interface.

We have created a live, interactive visualization using front-end web technologies and libraries jQuery, d3.js with SVG and JavaScript, combined with HTML/CSS. We parse the Arduino data directly from a log that is updated in real-time with any changes in the Arduino hardware/software system.

The Arduino aspect of the visualization displays students' use of the components, logical blocks and manipulations in software. At the time of writing this deliverable, we focus the visualization on when the students started using a given component, when they stopped using it and when they operated on it - links connected or disconnected between pieces of hardware or manipulations of settings in software. The visualization module includes a time-based interface where users can choose how much of the streaming data they want to see at full scale (see figure 67). In addition, we consider the integration of a summary module, which will display the amount of use of a given component, vs the total amount of time spent working at the PELARS workbench / classroom session. The latter visualization module will require integration with a face recognition module from the Computer Vision system.

The concept described above is a work in progress and has been iteratively developed within several prototyping sessions, as well as orientation and feedback rounds with learners and educators in different real world hands-on learning scenarios. The experiences, learnings and insights from this process will be reported in detail in D2.3 (due M22). The specifics (both technical and conceptual) of the integration of visualization modules within the IDE vs. (or in parallel to) a more complex, detailed stand-alone visualization application is yet to be

LEARNING DASHBOARD

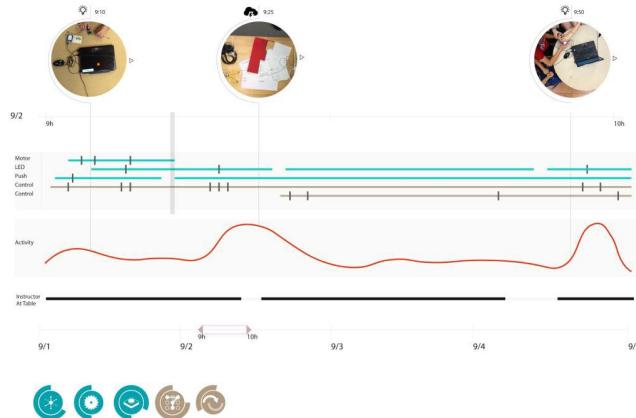


Figure 67: A screenshot of the visualization dashboard prototype, currently under development. The visualization module, reflecting usage of hardware components and logical blocks is laid out along a time of the current learning

explored and will be discussed in the upcoming months of the project. These developments will also include conceptualization and prototyping of visual modules that reflect more complex inter-relationships between various data streams of the learning process, among which 1) comparative visualizations regarding time spent actively working with the hardware vs. looking at the screen, to understand differences in work patterns, 2) collaborative patterns, discerned from both internal tracking of Arduino components, saved data that denotes individuals vs. groups and non-digital learning object coupled with gestural and action recognition measures, 3) punctuations and changes in work flow due to teacher intervention, etc.

Akin to the agile, iterative approach adopted by the project so far, these developments will be again prototyped and tested within real world learning environments to obtain early insights, inform new design affordances, as well as detect possible issues and emerging behaviours. They will be reported in the second edition of the iterative prototyping deliverable, namely D2.4 (due M32). The final visualization and feedback system will be integrated and

evaluated within the Learning Analytics System as part of Task 5.6 and will be described in the corresponding deliverable.

Connecting the Kit to the Collector

The student's interaction with the Visual Arduino IDE is part of the analytics process performed by the PELARS system. For this reason this information needs to be sent to the Learning Analytics System (LAS) that computes, together with the other information sources, learning traces. In particular the Visual Arduino IDE is capable of logging the creation of blocks, the connection of blocks and changes of parameters. This information can be considered medium level, that is now as low as mouse motions, but nor is at the semantic level.

As the other information sources running at the student's desk the information is sent from the Visual Arduino IDE to the PELARS Collector, which is the software tool that mediates between the sources and the remote server. As it is known from D5.1 the Collector collects the information with the possibility to store it locally.

The Collector API receives the logging information via a websocket that is listening on the Collector. The websocket protocol is suitable because it is packet based and well integrated into the JavaScript-based part of the Visual IDE. The API of the Collector is simple a bridge that encapsulate IDE events and sends them away.

A different type of API is planned for the Standard IDE in which metrics of the interaction and coding needs to be computed.

Manufacturing Rounds

At the time of writing the hardware design team had created four different rounds of boards to fulfil the project's needs. The different rounds are shown in table 9 with a proper description of each one of them.

It is planned to continue making more production rounds in the last quarter of 2015 to support further trials as well as for making usability tests of the TALKOO kit and to disseminate the project among potential future users of the system.

Date	Ver	Description	#
2014-09-01	v1	Proof of concept prototype, used to shoot the TALKOO concept video as well as to run tests on the hot-plugging protocol. Only 5 modules produced (LED, button, potentiometer, generic, MOSFET) as well as a programming device that allows having a smaller form factor for the modules.	10

2015-07-31	v2	Prototype round for trials during 2015 and 2016. 12 modules and a hub as described earlier in this deliverable. 5 kits manufactured by hand and distributed to different partners for testing and concept development.	65
2015-08-31	v2	Production finished on time for Ars Electronica 2015. It consisted of 16 extra modules of two types: buttons and RGB LEDs to complement the previous series. It was manufactured using a local facility in Malmö, Sweden.	16
2015-10-08	v2	Kit production for the trials. It was a series of 50 TALKOO kits to be distributed among partners. For this round normal LEDs were removed in favour of making more RGB LEDs and buttons.	650

Table 9: description of the different production batches of TALKOO kits, including prototype series

6 CONCLUSION

This deliverable is an overall status report on the technical development and kit design made within the PELARS project prior to launching a series of trials with students within three different learning scenarios.

The text describes the work that has been achieved in making:

- a fully functional hot-plugging modular electronics system with an initial design of 12 different modules and one so-called hub to connect the modules to a computer
- a cross-platform (working in Windows/Linux/OSX) visual programming environment to describe the relationships between modules by simply drawing lines between them
- a set of crafting materials to be augmented with the above-mentioned electronics for the students to perform a series of learning activities
- a series of learning activities for different learning scenarios
- a visualization tool to help students, teachers, and researchers highlight the

state of the learning analytics system, the

- over 700 boards manufactured in different ways: some by hand, some at different manufacturing facilities; these boards will feed the needs of WP7 in terms of equipment needed to run trials
- a plan for disseminating the use of the TALKOO kit and the information visualization tools outside the initial plans within the project
-

At the time of writing, ARD was in the process of testing the latest manufacturing round of 650 modules just on time for the first trials planned by MAH at Cibilab in Spain. It should be mentioned that we managed to accelerate the design and manufacturing process by doubling the design team involved at ARD. While our initial plans anticipated that we should have had kits ready by December 2015, we managed to have hand-made kits ready to be shown at Ars Electronica by July in the same year and trial-ready machine mounted kits in October.



APPENDIX D, PUBLICATION 1

Appendix D, Publication 1

D. Cuartielles, A. Göransson, T. Olsson & S. Stenslie. Mobile haptic technology development through artistic exploration. *Haptic and Audio Interaction Design* (Berlin: Springer-Verlag, 2012a), 31–40.

Reprinted/adapted by permission from Springer-Verlag. *Haptic and Audio Interaction Design* by Magnusson, Charlotte, Szymczak, Delphine, Brewster, Stephen © 2012

Mobile Haptic Technology Development Through Artistic Exploration

David Cuartielles, Andreas Göransson, Tony Olsson, and Ståle Stenslie

Medea, Malmö University, Sweden

david.cuartielles@mah.se

K3 - Malmö University, Sweden

{andreas.goransson, tony.olsson}@mah.se

Faculty of Humanities, Aalborg University, Denmark

stenslie@hum.aau.dk

Abstract. This paper investigates how artistic explorations can be useful for the development of mobile haptic technology. It presents an alternative framework of design for wearable haptics that contributes to the building of haptic communities outside specialized research contexts. The paper also presents our various wearable haptic systems for mobile computing capable of producing high-order tactile percepts. Our practice based approach suggests a design framework that can be applied to create advanced haptic stimulations/situations for physically embodied interaction in real-world settings.

Keywords: Applied haptics, wearables, bodysuit, haptic and embodied interaction, haptic resolution, Arduino, Android, mobile haptic systems, online haptics editor.

1 Introduction

This paper presents several of our artistic developments using mobile haptic technology with multiple tactile outputs (16+). These represent low cost, open-source haptic systems that use off the shelf components. Our approach intends to act as toolsets for designers working with haptic systems that create emotional and immersive haptic experiences. Rather than developing customized systems aimed at specific tasks or purposes we have made a set of “modules” that are tied together with a shared communication protocol. This approach allows for quick high-fidelity prototype development and faster, simplified iterations of the design. Since our projects are primarily based on standard components, they can be developed at a low start-up cost and propagate reusability.

Our projects have progressed in a chain of iterated design processes where the hardware and the conceptual components have affected each other. The conceptual content part of the system is based on an experimental media art approach where the goal is to create a multisensory, immersive and embodied experience system centered on an open exploration of a *poetics of touch*. The term embodiment is here understood as a combination of both a physical presence in the world and a social embedding in a

web of practices and purposes [1]. The resulting systems have been successfully tried at several usability tests during public art events in Norway, Sweden, Denmark and Slovenia.

1.1 Haptic Systems History

In history the concept of haptic communication through cutaneous touch can be traced back to Giovanni Battista della Porta who in 1558 described the sympathetic telegraph [2]. His proposal was to use magnetism to send and receive the same message over distance, encrypting and decrypting messages by tapping on to the body of two users. This rather imaginative device has never been built, but the concept represents an interesting first approach towards personal, direct and embodied corporal connectivity.

Other early concepts involving touch was Edison's '*Telephonoscope*' [3] which preconceived a telepresence system much like the later videoconferencing systems of today. An important inspiration for telepresence is the notion of being present at the other end of the communication line, as if one was physically present, sensing and interacting with one's own body.

One early important work on tactile interface technology was Bach-y-Rita's first 'tactile display' built in the 1960s [4]. A 20-by-20 array of metal rods in the back of chair were wired to act as the pixels of a screen and functioned much like an electronic Braille writer continuously raising and lowering 'dots' recognizable by the tactile senses. With this tactile display people sitting in the chairs could identify 'pictures' as they were poked into their backs. In effect this demonstrates cross modal perception, allowing us to see images with our sense of touch [5].

Creating a sense of tactile immersion through tactile manipulation of the senses is still difficult to invoke. Current haptic systems within areas such as telemedicine, telerobotics, CAD/CAM and virtual reality are primarily desktop based using technologies such as the PHANToM¹ (Personal HAptic iNTERface Mechanism) [5] and haptic gloves [6]. Most haptic interaction systems are based on a desktop paradigm [7]. This also goes for high resolution wearable displays such as the Tactile Torso Displays [8]. In our world of emerging smartphone- and mobile computing for users on the move, we foresee the need of wearable systems with a higher degree of mobility.

1.2 Towards Mobile and Wearable Haptic Systems

Today there exist no standard or commercially available systems for complex, high resolution haptic interfaces dedicated to mobile and wearable use. At the same time users are adapting to simple haptic systems such as vibrating screens and mobile phones, indicating both the growing need and possibility for somatosensory and haptic systems in communications and experience design. In later years wearable computing has become an extension of ubiquitous computing. This post-desktop, user-

¹ <http://www.sensable.com/products-haptic-devices.htm> accessed on June 7, 2012.

centric paradigm of human-computer-interaction focuses on embedding computational power seamlessly in everyday objects [9]. York also refers to it as machine fitting into the human environment [10]. By fitting haptic technology onto and into our bodies we can provide mobile users with information that was previously unavailable [11], one example being site-specific information and sensing related to users position (GPS) and orientation. For example the '*FeelSpace*' belt enables its user to feel his/her orientation in space via vibrotactile stimulation [12]. Such wearable systems are within a wearer's intimacy zone and therefore also have the potential to provide novel and highly expressive forms of interactions. An early example of wearable, haptic bodysuits, albeit attached to desktop computers, is the '*cyberSM*' system from 1993 [13] that connects two users over the internet allowing them to see, hear and touch each other.

One of the first mobile and telehaptic art projects was the '*Mobile Feelings*' project (2002-03) by Christa Sommerer & Laurent Mignonneau [14]. Here two users each held an egg shaped communication interface that let the users exchange heartbeats. The haptic effect was created with only one vibratory output, but still let users 'feel a strong sensation of bodily connection' [15]. They also note that 'the sense of touch still remains one of our most private sensation for which we still lack a concise language to describe'. However, as a language of touch appears contingent on haptic resolution² [13], it is likely that the minimal haptic resolution of one vibrator influenced the lack of haptic expressivity.

The '*Hug Shirt*' by the CuteCircuit company [16] attempts to construct haptic communication for simple, personal messages between users wearing what appears to be a normal looking shirt. The shirt transmits 'hugs' to another, similar shirt via a Bluetooth and Java enabled telephone device. The stimulus resembling a hug is produced by vibrotactile stimulation. Although scarcely described the shirt apparently has a haptic resolution of 10+ effectors. The company has worked on developing a taxonomy of hugs, but its effects are unclear. Another similar project is the '*Huggy Pyjamas*' by Cheok [17] that exchanges hugs through pneumatic actuators, allowing stronger sensations, but on the cost of wearability.

Thecla Schiphorst has worked on developing 'Semantics of Caress' [18] that investigates how the meaning of touch can be applied to tactile interaction. This system represents touch and movement as something meaningful, contributing to quality sharing. Having identified intrinsic values of haptic communication in systems with relatively low haptic resolution, one of our research questions has been how this can be translated into functioning, wearable systems that produce a greater degree of tactile immersion? High fidelity haptics implies a haptic resolution of 90+ effectors/actuators [13].

Usability issues such as weight, volume and power consumption poses a serious challenge to future system designs of wearable, mobile haptic systems. Lindeman et al.'s research [19] on full-body haptic feedback through applications made with their '*TactaVest*' haptic feedback system attempts to complete the user's sense of tactile immersion in a VR-based environment. The resolution of the early *TactaVest* physi-

² as the number of stimulators over surface of stimulation

cally confirms the haptic vision [5] experienced through VR. However, with only 8+ vibrotactile effectors it does not appear to have a high enough haptic resolution to provide a sense of sensory immersion on its own.

2 Artistic Research Methodology

The development of our haptic system has followed the path of affective interaction design where key aspects of the process are to effect emotional responses in the target user [20]. Emotional experiences do not solely reside in our minds or brains. They are experienced throughout our whole bodies [21]. Emotions have a crucial role in the human ability to understand and learn new things. Objects that are aesthetically pleasing appear to the user to be more effective by virtue of their sensual appeal [22].

Our research into affective haptics is grounded theoretically on practiced-based artistic research that is formed by the practice of making art [23]. Such artistically guided research is integrated in our projects through the construction of different practical-aesthetical experiments. Our various projects represent empirical research through testing prototypes of mobile, haptic interfaces. The advantages of building prototypes are many [24]. First of all it facilitates testing conditions that are not covered by established principles of design. Second, it provides an evaluation of a first concept for user interface as well as giving quick feedback from the user(s). Drawbacks include the temporary and limited experiential construction of prototypes. The scope of our prototypes has been to cover specific and aesthetically relevant aspects of technologically produced touch.

All our experiments have an open, explorative character, addressing the affective dimensions of haptic experience. How is this useful in scientific contexts? According to Schön, artistic works can be seen to represent knowledge, and the way the artist makes them reflects artistic methods [25]. We see making artworks as an integral part in building systematic knowledge about the use and application of haptic experiences. Our combination of affective interaction design approach with artistic exploration attempts to provoke emotional experiences in the user target group to reveal areas of problems throughout the different iterations of the hardware design.

Throughout our projects we have systematically applied a combination of the following methods: i) artistic practice-based research, ii) user interviews and iii) user observation. These were applied as tools to systematically create knowledge relevant to our goal of gaining insight and knowledge about haptic stimulus in mobile settings. The complex and multifaceted character of practical-aesthetical experiments demands bricolaged and interdisciplinary methods, therefore the use of ‘hybrid methodology’ is also suitable to describe our research

2.1 First Generation Mobile Haptics

Our first collaborative project, World Ripple, used GPS coordinates and satellite based navigation to create ‘immaterial sculptures placed in the open outdoor landscape. The sculptures are either location based events, a kind of haptic theatre, or

dynamic (data) structures moving, changing, developing their dimensions and properties over time' [26]. Users wore a bodysuit controlled by a GPS enabled laptop which connected to a micro-controller board and a custom made extension board called '*the dusk*' made for controlling the 64 coin-shaped vibrators inside the bodysuit. The bodysuit is worn underneath the ordinary clothing and the portable, sensor- and GPS based system is carried in a shoulder bag together with a laptop computer which controlled the entire system.

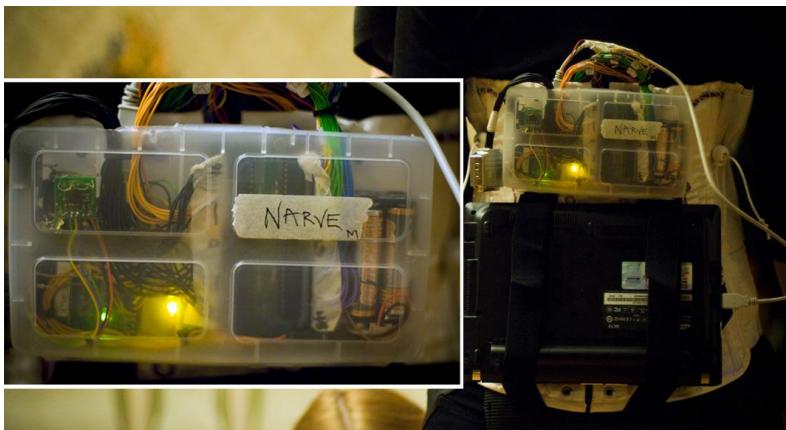


Fig. 1. Controller box with DUSK board (64 actuators) for BlindTheatre Bodysuit

The next iteration of the system was named 'The Blind Theatre' and turned the body into the stage of a somatic theatre (Fig. 1 and 2). It was performed indoors in 2009 at the Norwegian National Theatre in Oslo³. Users experienced an audio-haptic story while walking blindfolded around in the theatre. They reported high-order tactile percepts such as stroking, rubbing and caressing [27].

1. The blind theater project. <http://blindtheater.wordpress.com>



Fig. 2. The bodysuit for the Blind Theater. The haptic suit had two functional layers, one inner corsette and an outer, looser-fit cape.

2.2 Second Generation Mobile Haptics

The third iteration of the design replaced the laptop with the use of smartphones. This increased both comfort and maneuverability due to the smartphones weight and small form factor.



Fig. 3. The Psychoplastic bodysuit during use in Ljubljana 2010.

Much like World Ripple this system used GPS to geotag haptic data into zones defined by spatial coordinates. When entering predefined zones, users would trigger and sense the space through different combinations of vibrotactile patterns and sound (voice).

The phone both acted as GPS receiver and also ran the application controlling the vibrotactile outputs. The data was sent via Bluetooth to a custom made hardware named “Leaf” which consisted of a communication and control board extending into 5 boards shaped like flower leafs where each board controlled 16 vibrating motors, totaling 80 high-resolution actuators.

2.3 The Sense Memory Experiment

In the next Sense Memory experiment (2011) we simplified the system in order to experiment with an outdoor theatre involving multiple users. Constraints were cost and reliability. Using the new Arduino Mega board to Android system, the body suit contained only 16 high-resolution actuators. Although the haptic resolution was significantly lower, it simplified the first iteration of development and allowed us to prototype new hardware solutions within hours, focusing on maximizing the haptic effects outputs and design of haptic patterns. This also simplified the construction of the bodysuit, allowing us to – if needed – rapidly produce multiple suits.



Fig. 4. The Sense Memory Cape during the Malmö test.

Shaped as a cape (Fig. 4) the final bodysuit was designed for all-weather, outdoor use. Also here the user experienced invisible ‘sculptures’ by walking around in selected areas. New sets of haptic sculptures/expressions were geo-tagged onto 30+ zones placed around the square. Once the user enters one of the invisible geographic zones the sculptures came alive inside the cape as a combination of binaural sounds and vibrotactile patterns. Every zone contained different words and poetic expressions about emptiness in combination with unique haptic patterns. Here the user’s walk formed combinations of touch and words into a unique somatosensory story.

2.4 User observation and analysis

As part of our artistic research user feedback were collected through conducting informal interviews with several participants throughout all the iterations of the system. In combination with our own observations we present the following analytical results:

1. Immersive closure of space: Walking around in the public square during normal daytime activities, users reported they were mindful of it beforehand, but once inside one of the suit they quickly immersed into the experience and forgot about possible onlookers. This indicates a closure of space, strengthening users' sense of an intimate, personal and 'inner' experience.
2. Multimodal strengthening of senses indicating the affective roles of haptics and audio in interaction: the cross-modal combination of sound and touch was experienced as intermingled, intertwined into a mutual strengthening of stimulus. Most users reported focusing mainly on the sound heard and that this appeared as the strongest stimulus. However, they also commented that the touches experienced made them stay longer, thus intensifying the overall sensation of body and space.
3. Increase of spatial awareness: a higher degree of spatial awareness was both observed and reported. Users wearing one of the systems noticeably changed their movement in space, becoming more aware of how they were moving to find both new and previous zones of experience.
4. Behavioral change: as users were free to move around in the open space we possibly expected a systematic, grid like search for the various interactive areas. However, once they entered the first interactive area they tended to move slowly before stopping. Thereafter they were observed to move in what can be described as an irregular, search like manner, moving back and forth, turning back to previous zones. The quick adaptation to new movement and behavior indicates how easily users can adapt to haptic technologies.

2.5 Discussion

There are both benefits and problems with using an artistic exploration approach to the development of a wearable haptic system. Benefits compared to a traditional engineering approach that would focus on low level powering issues, long battery lives, or similar, is the artistic based strengthening of focus on the experiential dimension and user experience. Haptic and content related issues so become much more apparent in the early stages of development. In this way we can be more effective in creating valuable user experiences in faster design iterations. Such findings include positioning and repositioning of vibrotactile effectors in relation to sound and sequencing of haptic patterns. Focusing on users' sense of embodiment and immersion we found that anything less than 64 high-resolution haptic actuators seem to reduce the strength of the experience. When compared to other projects using higher densities of high resolution, vibrotactile stimulation such as Erotopod [13] a preferable number for a full bodysuit covering most parts of the body would be 90+.

One major challenge is how to develop and design haptic patterns that allow for high-order tactile percepts. Although not treated specifically here, our projects have designed iterations of haptic editors that facilitate rapid coding and testing of advanced haptic patterns. In combination with our wide range of body suit designs this allows for optimal actuator placement and combinations on the body.

Other challenges connected to the use of an artistic exploration approach relate to the evaluation of user feedback. In the nature of interactive art projects, the user be-

comes an integral part of the emotional experience they are evaluating. Though observations were made on changes in behavioral patterns during tests, assuring their scientific relevance is difficult since the users could not, in most cases, confirm or deny these changes as they were both performing an art project and at the same time being the actual art piece.

3 Conclusion and future developments

This paper contributes to i) the construction of functional, wearable and haptic experience systems and ii) the discourse of how actual embodiment is experienced within human computer interaction. The combination of i) hardware development and ii) conceptual, aesthetical work has greatly helped us develop new scenarios and novel approaches to the field of haptics. Our field tests show how geotagging haptic experiences greatly affects the users embodied experiences such as sense of place. Another outcome from our experiments is the suggestion that the experience of full body haptic immersion need a haptic resolution of 90+ effectors. There are many improvements to be made and future developments need to include high resolution and wearable prototypes capable of producing advanced haptic experiences for users on the move. Another aim is setting up a more rigid frame work for the evaluation of the artistic exploration approach to strengthen the academic relevance of the highly subjective results such artworks produce.

4 References

1. Dourish, P., *Where the action is: The Foundations of Embodied Interaction*. Cambridge, MA: The MIT Press, 2001.
2. Barnouw, E., *Mass communication: television, radio, film, press: the media and their practice in the United States of America*. New York: Rinehart, 1956.
3. Grau, O., *Virtual Art – From Illusion to Immersion*. Cambridge, MA: The MIT Press, 2004.
4. Zielinski, Siegfried (2002) Archäologie der Medien. Rowohlt Taschenbuch Verlag.
5. Paterson, M., *The Senses of Touch: Haptics, Affects and Technologies*. Oxford, UK: Berg, 2007.
6. Fong, B., Fong, A.C.M., Li, C.K., *Telemedicine Technologies: Information Technologies in Medicine and Telehealth*. West Sussex, UK: John Wiley & Sons, 2011.
7. Dominjon, L. Perret, J. Lécyer, A.: Novel devices and interaction techniques for human-scale haptics. In: *The Visual Computer*, vol. 23, pp. 257-266. Springer, Heidelberg (2007).
8. Van Veen, H.A.H.C. & Van Erp, J.B.F.: Providing Directional Information with Tactile Torso Displays. Presented at Eurohaptics 2003.
9. Weiser, M.: The computer for the 21st century. In: *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 3, pp. 3-11. ACM, New York (1991).
10. York, J., Pendharkar, P.C., (2004). Human-computer interaction issues for mobile computing in a variable work context. *International Journal of Human-Computer Studies* 60, 771-797.

11. Marion, A., Heinsen, E., Chin, R., Helmsø, B.: Wrist instrument opens new dimension in personal information. In: Hewlett-Packard Journal (1977).
12. Nagel, S.K., Carl, C., Kringe, T., Märtin, R., & König, P.: Beyond sensory substitution - learning the sixth sense. In: Journal of neural engineering, vol. 2, pp. 13-26. (2005)
13. Stenslie, S. *Virtual Touch – A study of the user and experience of touch in artistic, multimodal and computer-based environments*. Oslo: Oslo School of Architecture and Design, 2010.
14. Sommerer, C., Mignonneau, L., Mobile Feelings –wireless communication of heartbeat and breath for mobile art. In: Proceedings of The 14th International Conference on Artificial Reality and Telexistence. ICAT, 2004.
15. Ibid (Sommerer & Mignonneau, 2004)
16. Seymour, S., *Fashionable Technology*. Austria: Springer-Verlag/Vienna, 2001.
17. Cheok, A. D., *Art and Technology of Entertainment Computing and Communication*. UK: Springer/London, 2010.
18. Schiphorst, T., soft(n): Toward a Somaesthetics of Touch. In: Proc. of the 27th international conference extended abstracts on Human factors in computing systems, pp. 2427-2438. ACM, Boston (2009)
19. Lindeman, R.W., Page, R., Yanagida, Y., Sibert, J.L. Towards Full-Body Haptic Feedback: The Design and Deployment of a Spatialized Vibrotactile Feedback System. In: Proc. of ACM Virtual Reality Software and Technology (VRST), pp. 146-149. 2004, Hong Kong, China (2004)
20. Picard, R.W., *Affective Computing*. Cambridge, MIT press. 1997.
21. Davidson, R. J., Scherer, K. R. and Goldsmith, H. H., *Handbook of Affective Sciences*. New York: Oxford University Press, 2002.
22. Norman, D. *Emotional Design*. New York: Basic Books. 2005.
23. Biggs, Michael & Karlsson, Henrik (2010) The Routledge Companion to Research in the Arts, page 126. Taylor & Francis.
24. Starý C. (1996) Interaktive Systeme. Software Entwicklung und Software-Ergonomie. Vieweg Informatik/Wirtschaftsinformatik.
25. Schön, D., *The Reflective Practitioner*. New York: Basic Books. 1983.
26. World Ripple. http://www.stenslie.net/?page_id=85
27. Gibson, J., *The senses considered as perceptual systems*. Boston: Houghton Mifflin. (1966).



APPENDIX D, PUBLICATION 2

Appendix D, Publication 2

D. Cuartielles, A. Göransson, T. Olsson & S. Stenslie. Developing Visual Editors for High-Resolution Haptic Patterns. *The Seventh International Workshop on Haptic and Audio Interaction Design*, 42–4 (Lund: HaptiMap, 2012b).

Developing Visual Editors for High-Resolution Haptic Patterns

Cuartielles, D., Göransson, A., Olsson, T.
K3 – Malmö University
SE – 20506 MALMÖ, Sweden
david.cuartielles@mah.se
andreas.goransson@mah.se
tony.olsson@mah.se

Stenslie, S.
Aalborg University
Aalborg, Denmark
stenslie@hum.aau.dk

ABSTRACT

In this article we give an overview of our iterative work in developing visual editors for creating high resolution haptic patterns to be used in wearable, haptic feedback devices.

During the past four years we have found the need to address the question of how to represent, construct and edit high resolution haptic patterns so that they translate naturally to the user's haptic experience. To solve this question we have developed and tested several visual editors

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – graphical user interfaces (GUI), natural language, prototyping.

General Terms

Design, Languages.

Keywords

Haptic language, haptic editors.

1. INTRODUCTION

A haptic vocabulary is a 'toolbox' containing different ways and methods for touching users. The earlier ErotoGod installation (2003) [6] used a two-way bodysuit that both sensed users through input and touched them through vibrotactile stimulators. Here three kinds of touch patterns formed its haptic vocabulary. These were i) the ground, basic patterns used in every part of the installations dramaturgy, ii) the designed and specific patterns scripted as specific parts or sequences of the dramaturgy and iii) the random patterns that were generated as response to user's touch. Patterns were made without a visual editor. To aid the making of specific touches the patterns were first laid out as drawings marking positions and movement before coded directly into the software.

Much like a textual language, a haptic language consists of a hierarchical structure where the lowest layer defines the most basic components of the language, for example the characters of a text. This most basic component is then defined in more complex structures that produce yet another level of understanding; in our textual example this would be words. Multiple second tier constructs are then combined into the communicated message.

In our visual editors we have tried to facilitate both for exact control of single vibrator outputs -analogue to characters- as well as the formation of combinations of multiple outputs that form higher level and meaningful haptic expressions analogue to 'words'. Haptic languages can be subdivided into the alphabetic using tactile clues to form actual words (Braille, telegraph) and the conceptual, symbolic and non-verbal attempting to form

meaning through emotions and embodied sensations (hand gestures and body language) [2]. In this project we focus on the symbolic expression of haptic language. Although less exact it is much faster and experienced more direct for most users.

2. RELATED WORK

In 1998 Fogg et al. developed the haptic entertainment device called HandJive [4]. While developing HandJive the authors also realized the need of a novel haptic language for their device which they named "Tactilese". Composed of three hierarchical units – Positions, Patterns, and Routines – Tactilese lets the user create haptic messages of varying complexity which are then passed on to another player holding a similar device.

The basic units of Tactilese, Position and Pattern, were simple for users to grasp quickly. However, the more complex Routines had a much steeper learning curve and consequently users had difficulty grasping them, but given enough time they had no problem learning and performing them.

Many other works within the field of haptic interactions consider haptic languages as something immediate, a synchronous communication between two, or more users through devices equipped with haptic actuators [1-3,9,10]. The immediate communication is often initiated by an input device and received, and acted upon, by an output device. An example of this is the InTouch project by The Tangible Media Group at MIT Media Lab. Two identical roller devices in separate locations both record and exchange their movements, thus enabling their users to directly sense each other's actions.

Similarly, haptic feedback patterns can describe the environment in novel ways [5], actively helping the user to make decisions based on the information acquired by the haptic pattern.

Through this previous research we find that haptic patterns can be recorded, and/or played, in two modalities.

- i. Through direct touch on the haptic input device which then interprets that interaction and controls a haptic output device, synchronously or asynchronously. In this construction the haptic input device can be the same as the haptic output device; as in the example of HandJive, TapTap, and Huggy Pyjama.
- ii. Through an indirect definition or recording of a haptic pattern which is synchronously, or asynchronously, initiated by the haptic device based on events; originating either from the device status or its context. In the example of Soundercrumbs the device had a statically stored low resolution haptic pattern which was activated by the user in a specific context as navigational aid.

The biggest difference between the two modalities is the freedom to define the haptic language, in i) the user is free to define the vocabulary and its meaning herself, in ii) the meaning is already defined for her.

3. HAPTIC PATTERN RESOLUTION

A user defined haptic language is highly dependent on the number of haptic actuators implemented in the system. Ideally the amount of haptic actuators should exceed 90 to achieve a deep enough immersion into, and correct interpretation of, the high resolution haptic pattern [6].

In our project we define the difference between low-resolution haptic patterns and high-resolution haptic patterns as the combination of the number of haptic channels and each haptic channel's inherent data-resolution.

The garments developed in this project each had a varying number of haptic channels, ranging from 6 (Sweet) and 64 channels (Blind Theatre). Each channel had a data resolution of up to 128bit allowing us to precisely adjust the strength of the vibratory output of each channel.

4. THE PROTOTYPES

In this section we discuss the three latest iterations of the visual editor for high resolution haptic patterns. We also introduce our next version of the editor, currently in development.

4.1 First Generation Editors

The Blind Theatre editor (2009) was heavily influenced by the DMX GUI's used by technicians at the theatre. Featuring a total of 64 haptic actuators, and with little to no natural interactions planned in the user interface this editor had a very high threshold for beginners. While the interface was very complex, it also offered a high degree of control for each haptic actuator, and therefore an overall control of the entire suit.

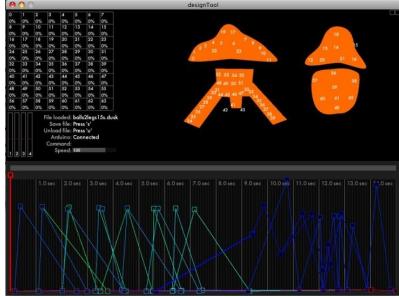


Figure 1 Blind Theatre Editor

4.2 Sweet Editor

Sweet marked the beginning of the second generation editors. Here we made a conscious move towards a much lower resolution haptic pattern compared to what was used in previous iterations, we settled with 6 channels because our aim in this project was not to explore haptic interactions but rather develop the concept of a haptic toolkit for designers.

The environment developed in Sweet should follow the basic design principles of user interfaces; among others it should be

clear, non-intimidating, and intriguing for the designer. Another important principle for Sweet was the one defined by Bret Victor in which he defines that creators need an immediate connection to their creations [8].

Therefore the visual environment of the Sweet Editor should:

- i. Resemble the outcome of the wearable device on which the high-resolution haptic pattern would be applied.

We envisioned that the Sweet Editor would allow the designer to insert a drawing or a photo of the garment giving the designer an immediate, and visual, connection between the garment and the haptic pattern. The Sweet environment should also:

- ii. Offer immediate connection to the outcome of the created haptic pattern.

Through the interface of the Sweet Editor the designer would be able of instantly installing, and testing, the pattern which is currently being edited. Another important concept of the Sweet Editor was also to move towards a more natural way of haptic interaction inside the programming environment – through touch.

- iii. The editor should be based on a touch-enabled platform allowing for a more natural interaction when creating the haptic pattern.

As there were no suitable touch-enabled platforms to sport the interface at the time of creation we tested functionality by mimicking touch-interactions through a standard WIMP interface. This enabled us to test basic functionality, but also caused confusion for test subjects.



Figure 2 The Sweet Editor

4.3 Sense Memory Editor

As cloud computing and streaming media is becoming the norm for accessing applications on mobile interfaces, we decided to move our tools and data online. This removed the need to actively install new geo-located high resolution audio-haptic patterns on the mobile device and instead issue a pull command to a web database when needed.

Also notable is that we decided to temporarily abandon the touch-based editing paradigm implemented in Sweet as that caused some confusion when applied to a WIMP style interface. The detailed time-line editor offered sufficient control for initial test, but represents a time consuming activity demanding several iterations of haptic pattern testing and re-editing.



Figure 3 Sense Memory Editor

4.4 Next Generation Editor

All our editors gain detailed control over the haptic channels in combination with an interactive timeline graph. However, as haptic stimulus must be edited one after the other, this represents both a slow and non-transparent approach. In realizing this we decided that our next generation editor should lessen detailed control over the haptic pattern through a timeline, and design towards a more natural touch interaction paradigm. Our new editor creates haptic patterns by touching iconographic representations of the body on a touch screen interface.

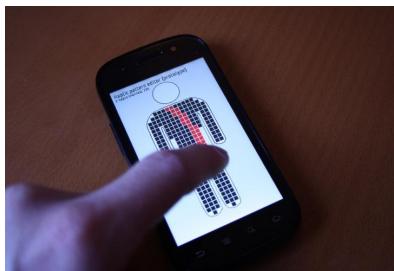


Figure 4 Next Generation Prototype

5. EDITOR ANALYSIS

The graphical editor iterations have been tested in several projects. Project group discussions and interviews have constructed the base for our iterations throughout the project, and from them we've gathered the following guiding principles for our next generation editor.

- i. The editor should support both modalities of creation and consumption of haptic patterns, and support this through both through asynchronous and synchronous communication.
- ii. Provide a direct feedback connection for designers working with the haptic pattern.
- iii. Direct feedback to haptic pattern generation on the small smartphone screens can be enhanced both through sound and vibration

- iv. Visual editors of full body haptic patterns should include iconography representing the whole body, both in front and rear view.

6. CONCLUSIONS AND FUTURE WORK

Developing visual editor software for high resolution haptic patterns and multimodal audio-haptic sculptures is a complex and difficult task. Possible solutions should focus on natural, touch based input to form and edit haptic patterns. Our intentions are to continue this project; developing visual editors for high resolution haptic patterns which are easy to use, portable between multiple systems and provide high-resolution haptic patterns in the three most common textual data formats – XML, JSON, and CSV.

7. ACKNOWLEDGMENTS

Our thanks to ACM SIGCHI for allowing us to modify templates they had developed.

8. REFERENCES

- [1] Bonanni, L., Vauelle, C., Lieberman, J., and Zuckerman, O.. 2006. TapTap: a haptic wearable for asynchronous distributed touch therapy. In *CHI '06 extended abstracts on Human factors in computing systems (CHI EA '06)*. ACM, New York, NY, USA, 580-585.
- [2] Chang, A. S., O'Modhrain, R. Jacob, E. Gunther, and H. Ishii. 2002. ComTouch: Design of a Vibrotactile Communication Device. In *Proc. ACM DIS 2002 Designing Interactive Systems Conference*. ACM Press.
- [3] Cheok, A. D., Choi, Y., Fernando, C. L., Peiris, R. L., and Fernando, O. N.. 2009. Huggy pajama: a parent and child hugging communication system. In *Proceedings of the 8th International Conference on Interaction Design and Children (IDC '09)*. ACM, New York, NY, USA, 290-291.
- [4] Fogg B.J., Cutler, L.D., Arnold, P., and Eishach, C. 1998. HandJive: a device for interpersonal haptic entertainment. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '98)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 57-64.
- [5] Magnusson, C., Bredegaard, B., Rasmussen-Grohn, K.: (2009) Soundscrubs – Hansel and Gretel in the 21st century. In *Proceedings of the 4th international workshop on Haptic and Audio Interaction Design (HAID '09)*
- [6] Stenslie, S. 2010. Virtual Touch – A Study of the use and Experience of touch in Artistic Multimodal and Computer-Based environments. Doctoral dissertation. The Oslo School of Architecture and Design, Oslo.
- [7] Tangible Media, Website at: <http://tangible.media.mit.edu/> accessed on April 7th 2010.
- [8] Victor, B., Inventing on Principle. Retrieved April 30, 2012, from Vimeo. <http://vimeo.com/36579366>
- [9] Wang, R., Quek, F., The, J. K. S., Cheok, A. D., and Lai, S.R. 2010. Design and evaluation of a wearable remote social touch device. In *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction (ICMI-MLMI '10)*. ACM, New York, NY, USA.
- [10] Wang, R. and Quek, F.. 2010. Touch & talk: contextualizing remote touch for affective interaction. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction (TEI '10)*. ACM, New York, NY, USA, 13-20.



APPENDIX D, PUBLICATION 3

Appendix D, Publication 3

D. Cuartielles, A. Göransson, T. Olsson & S. Stenslie. Telehaptic Awareness. In *Proceedings of the 7th conference on Tangible, embodied and embedded Interaction, TEI 2013*, 1–8 (Barcelona, Spain: ACM, 2013c).

Telehaptic Awareness

David Cuartielles

K3 - Malmö University
Östra Värvgatan 11a
211 19, Malmö SWEDEN
david.cuartielles@mah.se

Andreas Göransson

K3 - Malmö University
Östra Värvgatan 11a
211 19, Malmö SWEDEN
andreas.goransson@mah.se

Tony Olsson

K3 - Malmö University
Östra Värvgatan 11a
211 19, Malmö SWEDEN
tony.olsson@mah.se

Stahl Stenslie

Faculty of Humanities
Aalborg University
Frederik Bajers Vej 7F
9220 Aalborg, Denmark
stenslie@hum.aau.dk

Copyright is held by the author(s).
TEI 2013, February 10-13, 2013, Barcelona, Spain
ACM

Abstract

In this paper we present the next iteration in our study of wearable and mobile haptic communication, proposing to conduct a many-to-many haptic communication experiment over time. We wish to present the project and results in both the Demo and Poster format.

Keywords

Haptic, telehaptics, social haptics, haptic language.

ACM Classification Keywords

H5.2. Information interfaces and presentation (e.g., HCI): User interfaces.

Introduction

There are several projects that have investigated telehaptic communication between single users. However, there are few, if any projects that have investigated telehaptic communication between many-to-many users. Given the mass usage of smartphones and social media; such 'social haptics' promise to shed new light on technologically possible, future applications that employ embodied communication.

The project group has developed several single user, full body haptic systems with up to 80 variable vibrotactile outputs [1]. These represent advanced and specialized single user haptic systems. In this

experiment we propose to change focus toward a telehaptic system targeting multiple users. A critical issue for mobile haptic applications is that users physically have to wear the system on their body. Current full body systems are too complex to be worn over time [2]. A simplified system enables the user to wear and use haptic interactions more frequently. Reducing the complexity of our previous solutions enables us to investigate haptic communication and stimulation in an everyday setting and over longer periods of time. This study will help us understand more about the effects of haptic interactions in a many-to-many scenario where users are distributed over large distances; it will also work as the foundation for defining our future studies within distributed haptic interactions.

Telehaptic Communication History

There are several projects that have investigated telehaptic communication between single users. One of the first was The Transatlantic Telephonic Arm Wrestling system developed by Norman White and Doug Back in 1986. It was a simple force-feedback system that allowed participants in two remote locations to arm wrestle [3]. The technology was based on a motorized custom-built force-feedback system. Through remote modem connection the force exerted on the local electro-mechanical wrestling arm would be transmitted to the remote location, and vice-versa. It was shown in a transatlantic link between the Canadian Cultural Center in Paris, and the Artculture Resource Centre in Toronto. As a system it is interesting for the study of haptics in several ways. Firstly it is a real-time, two way system where the users can exchange brute force immediately, thereby gaining a good sense of presence as well as communication. Secondly, this

immediacy creates a shared and social event. It also demonstrates the importance of having 'real' players at each end. If the opponent was replaced by a computer it would function like a Turing test, possibly revealing a machine-like behavior over time. Relevant for this paper is how the project apparently causes a strong sense of telepresence with only one actuator device. However, its haptic resolution appears limited and the range of meaningful expressions is for that reason uncertain. A project related in technology is the 'Networked Neuro-Baby' by Naoko Tosa, demonstrated in 1995 between Tokyo and Los Angeles. Users could both communicate to the emotionally expressive visualization of the Neuro Baby software and remotely shake hands by squeezing a 'Handshaking Device'. This device measured the handshake's pressure and relayed the position and pressure data to the remote user through a force feedback interface [4], [5].

The first telehaptic system to connect two users wearing bodysuits was the cyberSM experiment. In 1993 it connected participants between Paris and Cologne [2].

One of the first mobile and telehaptic art projects was the 'Mobile Feelings' project by Christa Sommerer & Laurent Mignonneau [5]. Here two users each held an egg-shaped communication interface that let the users exchange heartbeats. The haptic effect was created with only one vibratory output, but still let users 'feel a strong sensation of bodily connection' (*ibid*). They also note that 'the sense of touch still remains one of our most private sensations for which we still lack a concise language to describe'. However, as a language of touch appears contingent on haptic resolution (as the number of stimulators over surface of stimulation), it is likely

that the minimal haptic resolution of one vibrator influenced a lack of haptic expressivity [2].

The 'Hug Shirt' by the CuteCircuit company attempts to construct haptic communication for simple, personal messages between users wearing what appears to be a normal looking shirt. The shirt transmits 'hugs' in the form of vibrotactile stimulation to another, similar shirt via a Bluetooth and Java enabled telephone device. Although scarcely described the shirt apparently has a haptic resolution of 10+ effectors. The company has worked on developing a taxonomy of hugs, but its effects are unclear.

Another similar project is the 'Huggy Pyjamas' by Cheok [6] that exchanges hugs through pneumatic actuators, allowing stronger sensations, but on the cost of wearability.

In the field of interaction design, the Strangely Familiar [7] project at Ivrea produced two projects that dealt with remote communication and use indirect haptic feedback. The Tok Tok by Aram Armstrong was a communication system connecting two distant lovers in faraway cities such as Tokyo, Toronto or Tel Aviv. When each lover knock on their box shaped interface, this is heard as a sonar like pulse in the box in the remote location. The receiving box then automatically 'ping's' the sound back to the original box. The time delay for the signal to return is based on the distance between the lovers locations. The further away, the longer the delay. In the other project, Tug Tug by Haiyan Zhang, dedicated and old style telephones offer an extra layer of interaction. By pulling the cord connecting the handset to the base on each phone, each person physically affects the other cord.

The Lega project [8] is a system used as a communication enhancer. It is made of a series of identical devices used in exhibitions. Lega devices are used in groups of 2-5 people. Each person carry their own Lega and moves around the exhibition freely. The devices, somehow egg shaped, provide haptic feedback resembling the movements made by users when experiencing a piece of art. Each device communicate wirelessly with all the other Legas in the confinement of the museum, replicating the movements by means of vibrations.

All these mentioned projects create some kind of dedicated telehaptic communication between two users. However, besides Lega there are few, if any projects that have investigated many-to-many telehaptic communication using wearable devices. Therefore, we propose to investigate how multiple users experience telehaptic interactions over extended periods of time and large physical distances.

The Telehaptic Awareness Project

State of the art of Smartphone technology allows prototyping new scenarios where users can be located almost anywhere and where they can wear the devices at any time of the day. Smartphones and the Arduino [9] system are lightweight and relatively transparent and can therefore be mounted/worn both ergonomically and aesthetically pleasing. To test how well this can be done we propose to build and explore a telehaptic experiment between multiple users and over longer periods of interaction. Based on the experiences from the above mentioned projects and our many haptic systems we want to gain more knowledge of possible focal areas in a many-to-many haptic interaction scenario. Several open research questions will be

explored. How does long term telehaptic communication affect and possibly increase our (social and physical) awareness of those we interact with? Indicators of this can be differences in the sense of (others) presence, multimodal interactions (touch in combination with voice, sms, mail, images) and more. Research parameters of importance in networking are [10] similarity between group members, incentives for communication (research in haptics), standards to types of inquiry and response (our own platform), facilitators for increased system functionality (iterative improvement throughout the project) and finally attitudes conducive to information exchange (all four participants have a stake in the research).

Experiment Setup

The four users will attempt to wear the system for up four weeks. There will be scheduled daily Skype meetings to monitor and register use and problems of use of haptic feedback. Throughout the test period we also expect several cycles of system improvement, in particular with reference to software, but also possibly hardware issues such as battery cycles/systems as well as modifications to the initial layout of the haptic 'necklace'.

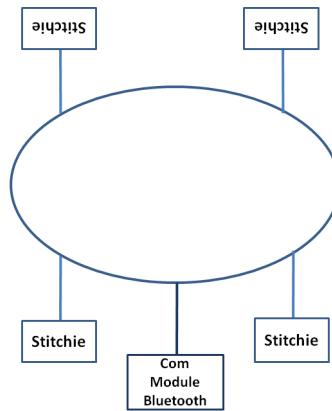


figure 1. Wearable haptic interface layout.

When applying touch in multiple user interactions it is necessary to identify who is doing what. We call such recognizable patterns for haptic identifiers. These are expected to contribute to the formation of meaning. At the outset there will be an open exploration of the systems. The experiments will start without individual identifications. Other open research questions this might answer are whether users over time create and leave identifiable, unique tactile 'fingerprints'? What are our behavioral haptic patterns? Who 'scratches' what and when?

System Setup

The work in progress setup involves four users. Each user wears a lightweight, wearable and mobile haptic input/output (IO) system shaped as a necklace and worn over the neck. The necklace has a tactile resolution [2] of four variable, vibrotactile actuators. Each actuator is called a 'Stichie' and is an addressable mini smartboard.

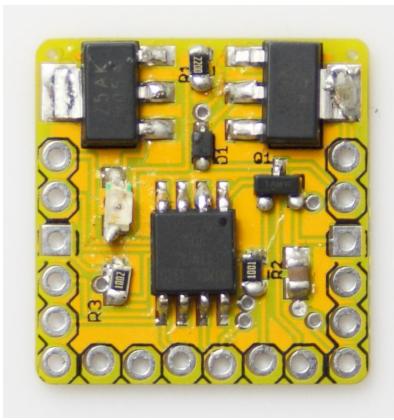


figure 2. Stichies module version 2, assembled.

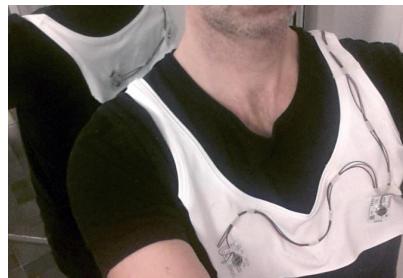


figure 3. First prototype of necklace

These Stichies are designed for easy application in almost any context. They allow for simple expansion of the system, even attaching multiple systems together, creating a very high maximum tactile resolution. The 'Stichies' have been tested in a one suit, one user environment [1].

The publish/subscribe paradigm, as described by [11], makes a lot of sense when developing a distributed haptic system like this because of the inherent space, time and synchronization decoupling between senders and receivers of haptic events. There are multiple publish/subscribe protocols developed, each intended for specific contexts.

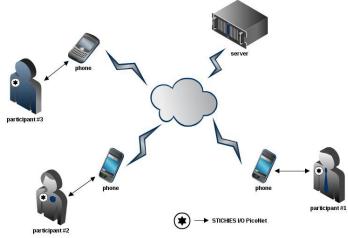


figure 4. Overview of the experimental setup.

The Message Queue Telemetry Transport protocol (MQTT) [12] is one of these protocols which are designed to be lightweight with a low overhead. Because of its small footprint it works well on low-bandwidth connections, such as mobile data networks.

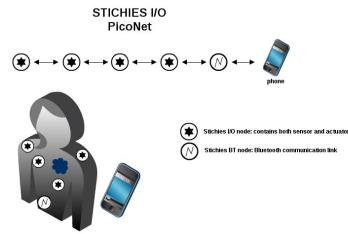


figure 5. The internal network diagram of Stichies.

Haptic Stimulus positioning

With only four variable vibrotactile actuators, it is vital that their positioning on the body is optimal and distinctly recognizable. To keep the system as simple as possible, both in terms of technical complexity and wearability, we have opted for two haptic zones on the back of the shoulders and two on the chest. All vibrators will be placed on soft tissue as not to reverberate through bones and thereby stimulate larger, unidentifiable parts of the body. Based on our initial 'necklace' prototype and testing, these four zones appear to be the optimal locations of vibrators on the body. The prototype with four Stichies producing variable, vibrotactile stimulus indicates that a range of haptic sensations and illusions [13] can be produced.

Haptic Language

Previous projects have a common difficulty in clearly describing what haptic sensations and illusions [13] can be produced and what perceived meanings they convey. How to understand tactile information in an intuitive and easy manner is a challenging question. Braille and Tadoma are examples of how it can be generalized, but on the cost of for example a steep learning curve [2].

The project will initially apply clearly distinguishable haptic sensations such as the so called TAU phenomenon that describes a tactile illusion of apparent movement. When a multiple point haptic system produces a series of short and discrete pressure sensations on the skin, they are perceived as movement between the points. Various patterns of vibrations over time promise to let the users form and distinguish between haptic messages. Another of our open research questions is what meanings are

perceived. How can haptic communication form a haptic language? In relation to this Thecla Schiphorst has worked on developing a 'Semantics of Caress' [14] that investigates how the meaning of touch can be applied to tactile interaction. Her approach represents touch and movement as something meaningful, contributing to quality sharing. Having identified vague, but intrinsic values of haptic communication in systems with relatively low haptic resolution, one of our research questions will be how this can be translated into functioning, wearable systems that produce a greater degree of tactile immersion? Here fore we will apply new smartphone technologies and open source Arduino hardware.

Methodology

The experiment will apply a qualitative research design mix of participatory action research and ethnomethodology. Action research methodology is useful as it involves the process of 'actively participating in an experimental change situation whilst the members conduct the research. Ethnomethodology is useful as the members will self-reflect on their participation in the exploration of the haptic communication and its phenomena. The result will be presented as a case report reflecting the knowledge and meanings that the four participants reported.

The setup will first be tested by four members of the research team. Advantages for this approach are, among other i) short prototype cycles, ii) fast feedback, iii) friendly familiarity between participants and iv) expected lower threshold to experiment with patterns.

aWearness

The combination of a communication system targeting increases in social awareness with a wearable haptic system can be termed an 'aWearness system'. A key indicator in the experiment is if users will report an increase in aWearability? That will be measured both as awareness of the other group members and transparency of the lightweight and invisible (to others) hardware. Other indicators we are looking for will range from awareness of others somatic sensations to their movement, state of mind, and emotions. As such as experiment has not been done before, we expect to uncover several other issues and questions as the project develops. Some of the questions that may arise through our study might be i) How is a haptic communication system perceived when worn over longer periods of time? (Steve Mann), ii) In a many-to-many haptic system, how are individual users identified?, iii) Who is touching what, and when?, iv) Will users develop unique touch patterns?, v) Can users be distinguished through 'haptic fingerprints?' and vi) How are haptic patterns influenced and changing due to contextual changes in everyday settings? Questions related to usability issues are for example if such a system is suitable in an everyday setting and whether the lightweight necklace is perceived as 'transparent'?

Expected Outcomes

We believe this Telehaptic Awareness project will bring in new knowledge about wearable haptic communication systems such as indications if it is sustainable to be used by a mass audience.

References:

- [1] Cuartielles, D., Göransson, A., Olsson, T., Stenslie, S., 2012. Mobile Haptic Technology Development through Artistic Exploration. In *Proceedings of the 7th International Conference on Haptic and Audio Interaction Design*, Lund, Sweden, August 2012.
- [2] Stenslie, S. *Virtual Touch – A study of the user and experience of touch in artistic, multimodal and computer-based environments*. Oslo: Oslo School of Architecture and Design, 2010.
[Http://virtualtouch.wordpress.com](http://virtualtouch.wordpress.com)
- [3] Norman, White T.,
<http://www.normill.ca/artpage.html> accessed on November 21st 2012.
- [4] Tosa et. al., Neuro-Baby,
http://www.tosa.media.kyoto-u.ac.jp/tosahp-old-20091202/nb/nb_paper.html accessed on November 21st 2012.
- [5] Sommerer, C., Mignonneau, L., Mobile Feelings – wireless communication of heartbeat and breath for mobile art. In: *Proceedings of The 14th International Conference on Artificial Reality and Telexistence. ICAT*, 2004.
- [6] Cheok, A. D., *Art and Technology of Entertainment Computing and Communication*. UK: Springer/London, 2010.
- [7] Strangely Familiar 2005. Exhibition at Interaction Design Institute Ivrea, 2005, video at:
<http://www.youtube.com/watch?v=7sqf8RSbuN8> and booklet at:
<http://projectsfinal.interactionivrea.org/2004-2005/Strangely%20Familiar%202005/presentation s/booklet/book01.pdf>
- [8] LEGA <http://www.mobile-life.org/> → The Lega: A Device for Leaving and Finding Tactile Traces - Jarmo Laaksolahti, Jakob Tholander, Marcus Lundén, Jordi Solsona Belenguer, Anna Karlsson, Tove Jaensson - gotta add the ACM reference instead of this → <http://www.borisdesignstudio.com/supple-interaction-lega/>
- [9] Arduino, <http://www.arduino.cc> accessed on November 26th 2012.
- [10] Stevens, Chandler H.. MANY-TO-MANY COMMUNICATION. CISR No. 72, Sloan WP No. 1225-81. Center for Information Systems Research. Sloan School of Management. Massachusetts Institute of Technology, 1981.
- [11] Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. 2003. The many faces of publish/subscribe. *ACM Comput. Surv.* 35, 2 (June 2003), 114–131.
DOI=10.1145/857076.857078
<http://doi.acm.org/10.1145/857076.857078>
- [12] The MQTT protocol definition
<http://www.ibm.com/developerworks/webservices/library/ws-mqtt/index.html>
- [13] Hayward, Vincent. A Brief Taxonomy of Tactile Illusions and Demonstrations That Can Be Done In a Hardware Store. *Brain Research Bulletin*, 2008, Vol 75, Nr 6, pp 742–752.
- [14] Schiphorst, T., soft(n): Toward a Somaesthetics of Touch. In: Proc. of the 27th international conference extended abstracts on Human factors in computing systems, pp. 2427–2438. ACM, Boston (2009)

Dissertation Series:

New Media, Public Spheres and Forms of
Expression

Culture and Society

School of Arts and Communication, K3

Malmö University



978-91-7104-942-1 (print)

978-91-7104-943-8 (pdf)

MALMÖ UNIVERSITY
205 06 Malmö, Sweden
WWW.MAU.SE