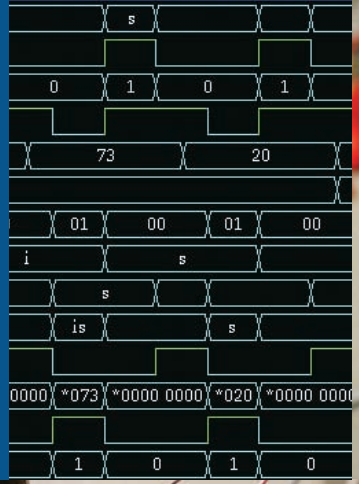
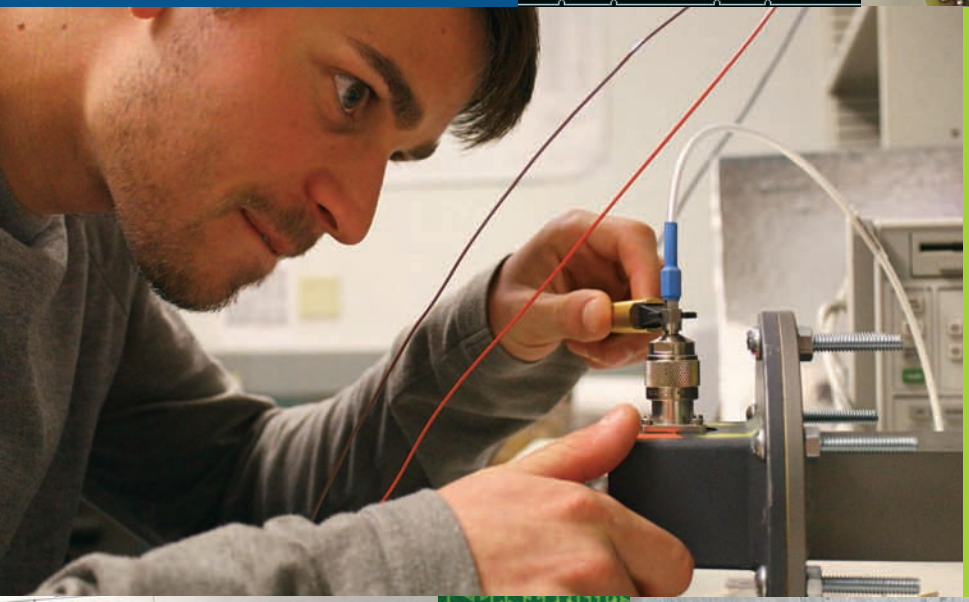
Baskin School of Engineering





2014

CORPORATE SPONSORED

SENIOR PROJECTS

PROGRAM

PARTNER'S DAY

Baskin Engineering UC SANTA CRUZ

INTRODUCTION

We are pleased to provide this booklet highlighting our third year of the Corporate Sponsored Senior Project Program also including this year's Capstone projects from our non-sponsored student teams in Computer Engineering and Electrical Engineering! Our students have worked very hard during their time at UC Santa Cruz earning their degree and fulfilling this capstone design sequence.

"[I]f students are to be prepared to enter new-century engineering, the center of engineering education should be professional practice, integrating technical knowledge and skills of practice" [Sheppard et al., 2009]. Students who have participated in this Corporate Sponsored program have been provided with a unique opportunity to experience working on real-world projects that involve design, budgets, deadlines, teamwork and reviews with their team mentor. They have come away with a sense of professionalism and pride in their work learned challenging skills, experienced entrepreneurship and been introduced to the many implications of intellectual property.

Throughout this academic year, the students have interacted with their teammates, some have made visits to their corporate sponsor's worksite and all have solved problems that arose along the way. The students take great pride in their completed projects and all they have accomplished during their time at UC Santa Cruz and the Baskin School of Engineering.

We also take great pride in what the students have accomplished. We are very grateful to our corporate sponsors for their willingness to sponsor this year-long program, mentor our students and provide them with challenging projects to work on.

Arthur P. Ramirez
Dean
Baskin School of Engineering

Sheppard, Sheri, Kelly Macatangay, Anne Colby and William Sullivan.
Educating Engineers: Designing for the Future of the Field, Jossey-Bass, 2009.

Baskin Engineering
UC SANTA CRUZ

ACKNOWLEDGEMENTS

We would like to acknowledge and thank the faculty and staff who have been so instrumental in the Corporate Sponsored Senior Project Program:

SENIOR DESIGN FACULTY

Patrick Mantey—Director, Corporate Sponsored Senior Project Program and Associate Dean for Industry Programs
soe.ucsc.edu/people/mantey

David Munday—Lecturer, Computer Engineering

Mircea Teodorescu—Associate Adjunct Professor, Computer Engineering
soe.ucsc.edu/people/mteodorescu

John Vesecky—Professor, Electrical Engineering
soe.ucsc.edu/people/vesecky

Linda Werner—Adjunct Professor, Computer Science
soe.ucsc.edu/people/linda

TEACHING ASSISTANTS

Jeffrey Bertalotto—Teaching Assistant

Paul Naud—Teaching Assistant

Ethan Papp—Teaching Assistant

CORPORATE SPONSORED SENIOR PROJECT PROGRAM STAFF

Lisa Coscarelli—Special Agreements Officer

Kimber Desmond—Project Program Assistant

Tim Gustafson—BSOE Technical Lead/BSOE Webmaster

Liv Hassett—Associate Campus Counsel

Frank Howley—Senior Director of Corporate Development

Heidi McGough—Program Coordinator/Executive Administrative Assistant, Dean's Office

Maureen McLean—Director of Resource Planning and Management

David Meek—Development Engineer, Baskin Engineering Lab Support

Christian Monnet—Development Engineer, Baskin Engineering Lab Support

Arthur Ramirez—Dean, Baskin School of Engineering

Lynne Sheehan—Network Administrator, Facilities/ Machine Room

Anna Stuart—Senior Budget Analyst

Bob Vitale—Director of Laboratories and Facilities

SPONSORS

SPECIAL THANKS to our sponsors for your generous support of our Corporate Sponsored Senior Projects Program. Your time, experience, and financial support were beneficial to our students and the success of their Senior Design Projects.

Capstone Project

Haptics Response Testbed

Stephanie Kwok, James Le, Thomas Moore, Stephanie Weber,
Stanley Wu, Nelson Zhou, Elizabeth Zitzer

Abstract

Haptics is the creation of tactile effects in devices using feedback systems. The Haptics Response Testbed project achieved this by the characterization of actuators, and the simulation of surface energy density. By evaluating the responses of actuator configurations in conjunction with the simulation models, the team has gained an understanding on how to manipulate the haptics performance within handheld devices. The project will allow companies to create a more varied and interactive tactile user experience for their customers.

Approach

Research was divided into two sections: hardware and software. The hardware team was responsible for characterizing eight piezoelectric (PZT) and linear resonant (LRA) actuators and evaluating the vibrational responses of a range of actuator configurations. The software team developed simulations of the test apparatus in LS-DYNA to verify the experimental results within reasonable error.

Stage 1 – Characterization

Hardware	Software
• Measure acceleration in testbuck using a variety of types and models of actuators in different positions	• Simulate the same tablet model in LS-DYNA and match properties and modal frequencies to hardware experiments

Stage 2 – Advanced Haptics

Hardware	Software
• Test multiple actuators out of phase or at different frequencies to characterize complex responses • Create waveforms to simulate a realistic button press	• Recreate the entire haptic system in LS-DYNA to examine possible combinations of actuators and positions. • Verify Z-axis acceleration with the hardware team and reach an acceptable threshold of error.

Acknowledgments

Ed Liljegren
Manager, Concept Engineering, Lab126
David Buuck
New Technology Integration Lead, Lab126
Ethan Papp
Corporate Sponsored Senior Design TA
Mircea Teodorescu
Assistant Adjunct Professor, Computer Engineering

Polytec
LSTC
Texas Instruments

Simulation

Figure 1. Meshed Testbuck Model

The software team created a 3D model in LS-DYNA and then performed modal analysis to examine the structural properties of the testbuck. Based on analytical results derived from "The Vibration of Thin Plates by using Modal Analysis" by Pouladkhan, the team calculated the natural frequencies of a plate of Teflon and then verified the results in LS-DYNA.

$$\omega_{11} = \pi^2\left(\frac{1}{length^2} + \frac{1}{width^2}\right)\sqrt{\frac{D}{\rho * height}} \quad (1)$$

$$D = \frac{E * height^3}{12(1-\vartheta^2)}, \quad \rho = density, \quad E = Young's\ Modulus, \quad \vartheta = Poisson's\ Ratio$$

Modal analysis minimized the error between analytical and simulated vibrational results. The team used the first mode (resonant frequency) to affirm their expectations regarding material properties and boundary conditions.

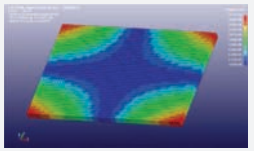


Figure 2. Modal Analysis (First Mode)

	First Mode of Testbuck	Second Mode of Testbuck
Analytical Results	223.66 Hz	430 Hz
FEA Results	160.54 Hz	191.04 Hz
Error	28.221%	55.572%

Table 1. Analytical vs FEA Results

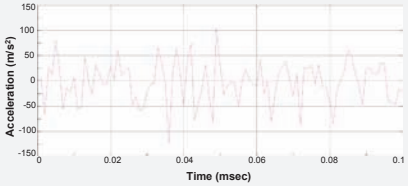


Figure 3. Center Node Testbuck Acceleration Analysis

The software team characterized a PZT and simulated the vibrational response produced in the Teflon testbuck. In Figure 3, the rigid body (average) acceleration of the testbuck is shown as the response to the characterized PZT vibrating at 220 Hz. The resulting vibration is shown below in Figure 4.

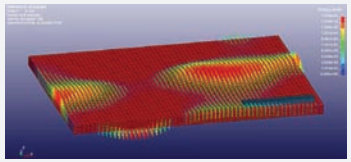


Figure 4. Vector Fields of Surface Acceleration

Experimentation

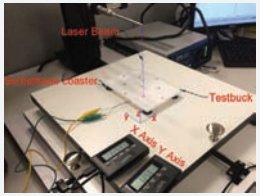


Figure 5. Laser Vibrometer Setup

The hardware team characterized eight LRAs and PZTs by varying the frequency, amplitude, duration, and envelope (FADE) of the input waveform used to drive the actuator. The vibrational response is measured in all three axes using an accelerometer attached to the testbuck. The hardware team verified the Z-axis measurements using a laser Doppler vibrometer (shown left).

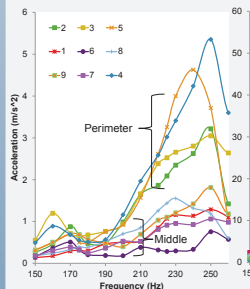


Figure 6. PZT Acceleration v Frequency

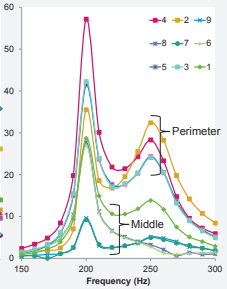


Figure 7. LRA Acceleration v Frequency

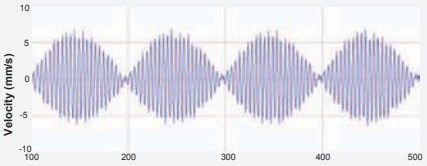


Figure 8. Beat frequency vibration from two PZTs at 230 Hz and at 240 Hz

The beat frequency test consists of varying the frequency difference between two actuators. One actuator is at a single frequency and the other increases by 10 Hz – 90 Hz above. This graph shows that the envelope of the output is at the beat frequency of the system. At low beat frequencies, the envelope is 100% modulated, where higher beat frequencies, modulation decreases until it is at 0% past 90 Hz.

Figure 9. Testbuck with node labels

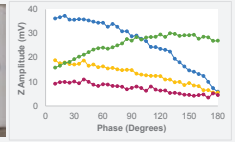


Figure 10. Phase Test Results

Figure 9 is a picture of the phase testing setup for the LRAs, and Figure 10 shows results of phase testing. From these we see that the further out of phase the input waveforms are, the more the amplitude decreases. The exception is the corner test, where the amplitude increased with phase shift.

Analysis

In varying the frequency, the hardware team found that PZTs have a wide bandwidth whereas LRAs have a narrow bandwidth. The PZTs actuate in the X or Y axis and so Z vibration is due to the energy propagating through the material. The LRAs actuate in the Z axis which makes their responses easier to feel. The shape of the input waveform is directly reflected in the resulting vibration, so to reduce audible noise, the waveform must ramp up in a sloped attack. The end of the waveform can be gradually decayed to reduce the testbuck's free vibration at its own natural frequency.

Actuating two PZTs at different frequencies results in a vibration whose waveform is modulated in amplitude at the beat frequency of the two input signals. This complex envelope is noticeably felt and could be a useful method of haptic texture creation.

Actuating two LRAs out of phase caused destructive interference in the testbuck, which reduced the amplitude response. When the two LRAs actuated at a 45 degree angle to each other, this effect inverted.

The software team's modal analysis and simulation of the 3D testbuck models verified the hardware team's experimental results. Comparing the results of Figure 2, showed a 28% error in the first mode. This is most likely due to the Teflon material damping the vibrations, not shown by the analytical results.

The results of the laser vibrometer reaffirmed the software team's simulation. Nodes along the sides of the testbuck displayed a larger acceleration compared to the nodes closer to the center. The amplitude of acceleration of the simulation is greater than the results of the vibrometer by twenty.

Results

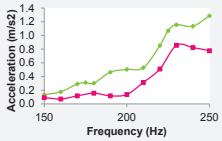


Figure 11. Acceleration of Accelerometer, Vibrometer, and Simulation

The amplitudes from the accelerometer, vibrometer and simulations behave similarly and differ in value by 20%, as shown in Figure 11. This accurate simulation can be used to create haptic events first in theory then easily translated into physical systems.

One example of a haptic event which can be applied in this way is a haptic button press. The purpose of the button press is to emulate the feel of an actual button on a mobile device. The waveform for the button press is shown in Figure 12 and it can be simulated in LS-DYNA and then recreated similarly on the physical system.

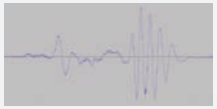


Figure 12. Haptic Button Press Waveform

Conclusion

For mobile applications, the PZTs have a faster response time, and a larger bandwidth, making them more suitable than LRAs . Advanced haptic responses can be emulated by varying either the phase or frequency difference between actuators. The team created a simulation which could accurately predict the acceleration at any point on the testbuck, and then verify that prediction with the physical hardware.

Looking Forward

This research is just the beginning for advanced haptic applications. The results will be utilized in consumer electronics such as mobile devices and gaming to create an enhanced user experience. Possible applications include safety features in cars, virtual reality gaming, medical devices, and e-readers.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Characterization of Glass at Millimeter Waves

Austin Kress, Jarred Moore, Alexis Rocha-Roux, Steven Telles

CORNING

Abstract

The aim of this project is to develop a reliable test method in determining the dielectric properties of a material at microwave frequencies. Testing the samples at frequencies of 20-100 GHz will lead to possible applications of the glass at 20-100 GHz.. Specifically we will be testing Eagle XG Glass and a fused silica glass manufactured by our corporate sponsor, Corning Incorporated.

Approach

The Eagle XG Glass has not been tested at microwave frequencies. To work around these unknown variables, we plan to compare two test methods at 3 GHz and 20 GHz..

The transmission line method.

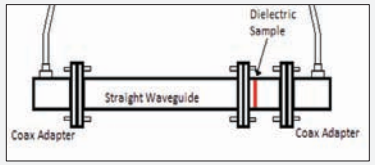


Figure 1 - Transmission line method experimental set-up.

The tapered transmission method.

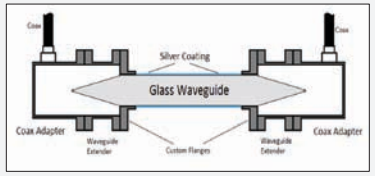


Figure 2 – Tapered transmission line method experimental set-up.

Overview

Designed tests to find the dielectric loss tangent of Corning glass samples in the S-band (2-4 GHz) and K-band (18-26.5 GHz) or more specifically microwave frequencies above 20 GHz.

Using the Vector Network Analyzer (VNA) we will be measuring s-parameters to find the complex permittivity [1]. The ratio between the imaginary and real parts of the complex permittivity [2] will yield to loss tangent of the material.

$$\varepsilon^* = \varepsilon' - j\varepsilon'' \qquad [1]$$

$$\tan(\delta) = \frac{\varepsilon''}{\varepsilon'} \qquad [2]$$

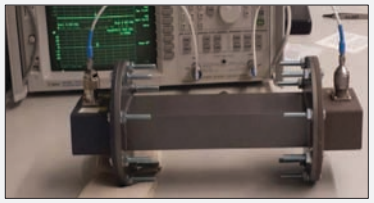


Figure 3 - S-band waveguide set-up for transmission line method.

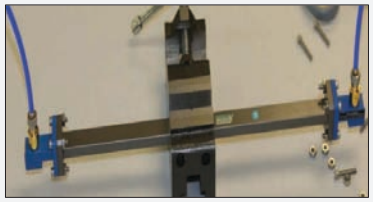


Figure 4 - K-band waveguide set-up for transmission line method.

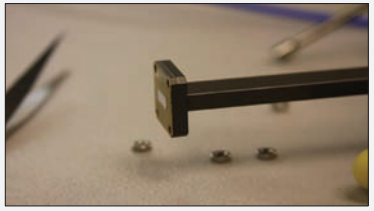


Figure 5- K-band waveguide with Teflon sample for transmission line method.

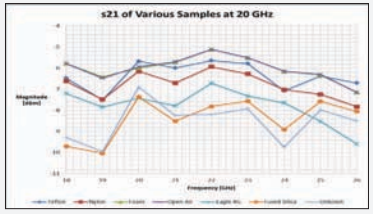


Figure 6- Magnitude of transmission coefficient at K-band frequencies.

Acknowledgments

Gary Trott, PhD Corning West Technology Center, Mentor/Sponsor
John Vesecky, PhD
Jeff Bertalotto
Frank "Bud" Bridges, PhD
Dave Thayer and Joe Cox, UCSC Machine Shop

Analysis

The transmission line test was performed by placing a thin sample into a waveguide and finding the difference between it and free space. We noted s_{11} and s_{21} that led us to calculate the loss tangent of the materials.

Large errors in the tapered transmission line test which led us to place emphasis of our project onto the transmission line method.

Results

Material	Loss Tangent	Dielectric Constant	Frequency [GHz]
Eagle XG (1)	0.762	3.04	3
Eagle XG (2)	0.502	3.687	3
Eagle XG (3)	0.692	3.196	3
Paraffin Wax	0.536	1.196	3
Teflon	0.013	2.45	20
Nylon	0.005	1.81	20
Fused Silica	0.054	0.535	20
Eagle XG	0.004	1.821	20
UNKNOWN	0.007	1.8054	20

Table 2 – Results from transmission line method.

Potential reasons for error:

- Calibration issues
- Imperfect glass (scratches, cracks, etc.)
- Imprecise sample dimensions.
- Cavity leakage
- Improper handling of samples (finger oils, etc.)
- Placement of samples within waveguide

Conclusion

The transmission line method seemed to work and we believe our other methods would have as well given we had exact samples requested. Unfortunately, this was more difficult than expected due to the precise dimensions we needed.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

HP Object Storage Metadata Search

Xiaoyuan Lu, Radhika Mitra, Masahiro Obuchi,
Nathaniel Rogers, TJ White

Abstract

Sponsored by HP, this project's goal is to add an Object Storage Metadata Search to OpenStack Swift.
The API allows a user to query the metadata of objects, containers, and accounts stored in the Swift system. The output is filtered according to user-specified query statements and attribute lists. The output can also be sorted and be presented in multiple formats in the HTTP body returned. The searchable metadata is stored in a database on a newly created server which is populated by crawlers. These crawlers periodically run on the Account, Container, and Object servers. This RESTful API is implemented in middleware on the proxy server.

Approach

The components of OpenStack Swift that our backend implementation communicates with are:

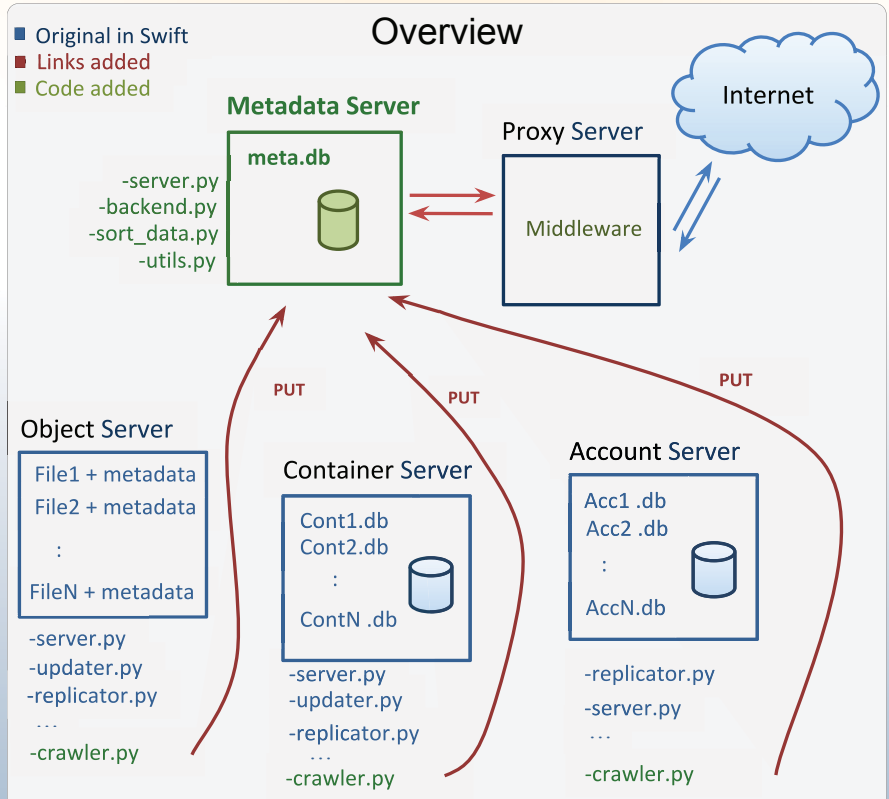
- **Proxy Servers**: The public face of OpenStack Swift, it handles all incoming API requests. Once a Proxy Server receives a request, it passes the request to the appropriate server.
- **Account & Container Servers**: Each Account and Container is an individual database that is distributed across the cluster. An Account database contains a list of Containers in that Account. A Container database contains a list of Objects in that Container
- **Object Servers**: Contains the data itself.

We add crawlers, or daemon processes, to the Account, Container and Object servers. These crawlers send the metadata to the Metadata Server which holds all of the requested metadata for all items in the OpenStack Swift cluster. The Metadata Server receives updates as the individual Objects, Containers, and Accounts are updated. The Proxy Server communicates with the Metadata Server over HTTP to fetch information and return the query's results.

Future Work

- Finish implementing whole specification
- Metadata Database Replication
- Security - Access control to metadata and Authorized Searchers Feature (from API)
- Improve efficiency of updating metadata
  - Middleware capture
  - Hook into update daemons
- Choose a more suitable database (SQLite not scalable)
  - MySQL? NoSQL? Distributed DB?

Overview



Interactions between the existing Swift architecture and our additions

Acknowledgements

Sam Fineberg, Distinguished Technologist, HP
Lincoln Thomas, Software Engineer, HP
Linda Werner, Faculty Advisor, UCSC
Michael McThrow, Teaching Assistant, UCSC

Crawler Structure

One of our design goals is to leave the OpenStack Swift base code the same for easy adding and removal of our crawler code. To accomplish this, we created a different crawler for each of these servers. These crawlers independently retrieve the metadata out of each server's backend and send it to the Metadata server. The crawlers run at a configurable interval. For Objects it only sends metadata if a change was detected within the last interval.

Server Structure

The Metadata server was added to the Swift architecture in order to have a database abstraction layer as well as an encapsulated Metadata Search API. The Metadata server has two main source code files:

"server.py" is the code that processes the Metadata Search API query. This code parses the query parameters and makes calls to the database abstraction layer to retrieve the requested metadata. It also receives and processes metadata sent from the crawler and stores by passing it to the database abstraction layer.

"backend.py" is the code for the database abstraction layer. We used SQLite3 for our implementation. We have four tables: Object, Container, and Account system metadata tables and a custom metadata table which has URI, key, and value columns.

Conclusion

The Object Storage Metadata Search API gives users greater insight into data stored in their OpenStack Swift Storage systems as well as providing an easy to use, concise, and powerful interface. The ability to query metadata provides a service that did not exist before and gives greater control into how a user sees the data stored in their cluster. The interface has many commands and specifiers that allow quick building of expressive queries.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Polymer Discovery

Thomas Goddard, Konstantin Litovskiy, Nathan Nichols-Roy,
Matthew Reed, Igor Shvartser, Nicholas Smith, David Zeppa

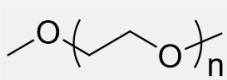
IBM

Abstract

The recognition of polymer images in the literature and patents is key for automatic understanding of the wealth of polymer data already known. While tools such as OSRA (Optical Structure Recognition Application) [1] exist to identify and interpret chemical structure images, these tools do not yet work for polymers. Our tool, Polymer OSRA(P-OSRA) extends OSRA's capabilities by being able to recognize brackets and parentheses in chemical diagrams. To date P-OSRA pre-scans the image; records and alters the image by removing brackets and parentheses from the diagram; calls OSRA and then collects and edits the resulting SMILES[2] string from OpenBabel[3] to reflect changes needed for describing a polymer image. P-OSRA then populates a data-model to allow for subsequent querying of polymer substructures (such as repeat units).

Introduction

Accelerated materials discovery is at the core of innovation, economic opportunities, and global competitiveness. The research process is responsible for bringing new materials to market. In 2011, President Obama launched the U.S. Material Genome Initiative (MGI) and challenged researchers, policy makers, and business leaders to reduce the time and resources needed to bring new materials to market—a process that today can take 20 years or more. There is great potential in leveraging modern data mining, big data analytics techniques, and physics based modeling (high performance computing) to significantly shorten the Research & Development cycle in material sciences. Polymers, as an important part of materials science, are the focus of much research for applications in the areas of semiconductors, nanomaterials, polymeric drug delivery vehicles, desalination membranes, and recyclable polymers for green chemistry. In order to accelerate polymer discovery, the first challenge is to automate the retrieval of all polymers made to date both in the literature and in patents. Although there are standardized nomenclatures for small molecules, standardized nomenclature for polymers is not straight forward and it is often more informative to represent these molecules as molecular structure diagrams or acronyms (e.g. PEG). Thus, this project's goal is to identify polymer images in published research, analyze them and output the structure to a data-model that allows their substructures to be queried.



Background

Optical Structure Recognition Application (OSRA)

P-OSRA extends OSRA, a tool for parsing images of chemical structures in documents, to support *polymer diagrams* that use bracket and parentheses notation in their representations. P-OSRA also offers a data interface to allow storage and querying of repeat units and end groups (sub-components of the polymers). This is important because

> "Proliferation of computer technologies has brought forward the necessity of new data formats to exchange information in a machine-readable way within the context of a scientific publication … Our approach to recovery of chemical information from published material is to reuse to the fullest extent possible the existing software created by the open source community and to invite further development and participation by releasing our work as free and open source ... OSRA has been designed with a wide range of applicability in mind: it does not rely on the document image being of any particular resolution, color depth, or having any particular font used. To manipulate images, OSRA employs the ImageMagick library [4] that allows parsing of over 90 different image formats, including the popular TIFF, JPEG, GIF, PNG, as well as Postscript and PDF…"[5, page 1].

Acknowledgement

Many thanks and appreciation for IBM Researchers Dr. Julia Rice, Dr. Hans Horn, and Dr. Amanda Engler; creator of OSRA, Dr. Igor Filippov; IBM Researcher Scott Spangler; and Professor Linda Werner, UCSC.

Results

High Level Process Diagram of P-OSRA

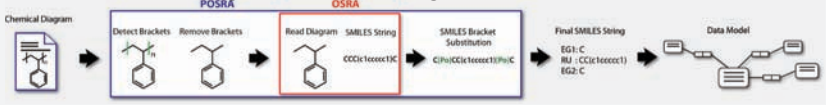


Figure 1

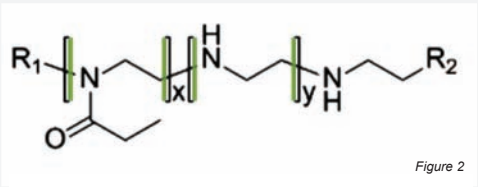


Figure 2

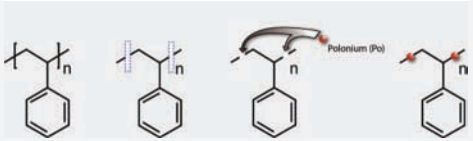


Figure 3

Finding and Removing Brackets

At first, we took the approach of using point density data from a vector conversion of a structure image to pinpoint segments of the diagram that could be brackets. This proved mostly inaccurate and difficult to implement. Next, we tried an algorithm to detect endpoints in the diagram and found this to be promising. This algorithm works by checking each pixel's surrounding adjacent pixels to see if it is an endpoint. We then look for pairs of endpoints that properly align, check for reasonable spacing and bond intersection, and tentatively store the coordinates of these endpoints as the location of a single bracket. From these coordinates, we draw a bounding box around the bracket and color all pixels inside the box white, removing the bracket, and then redraw the bond without the bracket. Figure 2 is the resultant image after tracing, in green, endpoints that are believed to be brackets. Figure 3 depicts the polymer polystyrene and the process in which we alter the image. After detecting the endpoints of the image and finding the brackets, we are able to box out the brackets entirely and replace them with placeholder atoms (we chose the bi-valent Po atom as it is highly unlikely to appear in any polymer structure).

Editing the SMILES String

Since we have replaced the brackets by placeholder atoms, OSRA then continues to parse the image as a normal chemical structure (red box in Figure 1). The resulting SMILES string (obtained by calling OpenBabel) reflects this structure. But since the image represents a polymer, the generated SMILES string needs to be edited by splicing it (at the locations of the Po placeholder atoms) and formatting it into end groups and repeat units. Figure 4 is a chemical diagram of a polymer poly(trimethylene carbonate) and it has been color coded to distinguish end groups ("EndGroups 1,2") from repeating segments ("RepeatUnit"). Figure 5 is the resulting SMILES string after OSRA/OpenBabel has completed parsing the diagram (top) and the final edited SMILES strings for the polymer (bottom).

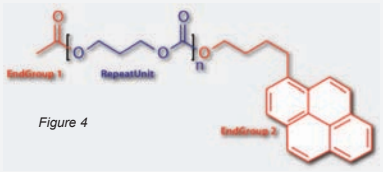


Figure 4

```
CC(=O)[Po]OCCCOC(=O)[Po]OCCCCc1ccc2ccc3cccc4ccc1c2c34

EG1: CC(=O)X1
RU:  X1OCCCOC(=O)X2
EG2: X2OCCCCc1ccc2ccc3cccc4ccc1c2c34
```

Figure 5

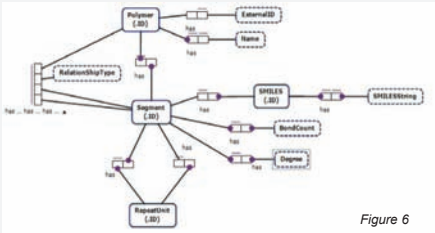


Figure 6

Creating a Data Model

We have designed a data model in Object Relational Model format (ORM)[6] so that we can store and query the images. With a queryable database, researchers can search for particular substructures or "similar" repeat units that they may desire to include in a new polymer. Retrieval of these units will allow them to rapidly assemble and test new polymers via computer simulation and eventually in the lab. ORM is very flexible and can output to many database formats including MySQL[7]. Figure 6 shows an ORM diagram tree, graphically showing the relationship between substructures and information about the polymer. For example, a Segment may have exactly one SMILES identifier, which has exactly one SMILES string, or in a more complicated example a Segment might have one or more RepeatUnits, that may or may not be nested within each other.

Initial Challenges

Understanding the Code

A publication by OSRA's creator, Igor Filippov, helped us to understand the basic function of the program giving us insight into which libraries we needed to be familiar with, but it did not reveal the underlying structure or details of the code. Because the OSRA program code uses many image processing and chemistry terms that we were not familiar with, we spent many hours in code reviews, compiling internal documentation, and generating a function tree using Doxygen[8]. Using these software engineering tools and techniques, we were able to identify locations for code insertion while preserving the functionality of OSRA.

Understanding the Fundamentals of Chemistry

Initially the team was given basic chemistry data sheets, which touched on electron density and orbital theory, to determine the chemistry knowledge the team had at the start of the project. From the questions and concerns of the team, it was clear that there were many inadequacies in the team's overall knowledge of organic chemistry including basic terminology. Thus, a lecture on nomenclature of organic families was given by Drs. Rice, Horn, and Engler to build the necessary organic chemistry knowledge base the team would need to use to communicate during the project (personal communication, January 2014).

Creating a Test Harness

One of our goals in building the P-OSRA extension was to show that individual features of the program worked correctly. Unit testing allows us to isolate each part of the program and find problems early in the development cycle. First, we collected data including a set of selected images and their corresponding SMILES strings that our university team and IBM advisors believed to be appropriate for testing. We then wrote test scripts to automate processing on the selected images. This process not only makes development more efficient, but it also helps us benchmark and optimize the code to preserve OSRA's power.

Performance

OSRA has been parallelized with OpenMP[9], allowing the program to be multithreaded. Since the program involves mostly image processing requiring multiple passes over images at different resolutions, OSRA lends itself to parallelization allowing batch processing to be more efficient. With our additions, we hope to preserve the same level of performance and plan to better incorporate OpenMP in our image pre-processing stage to maximize efficiency.

Conclusions

We have created an extension to OSRA, making it more comprehensive, that provides the ability to parse polymer chemical diagrams. Our plans include the addition of more comprehensive and detailed documentation and development tools to OSRA to make it easier for other teams to extend OSRA's capabilities. POSRA is written in C++ and is currently supported on Windows and Linux systems, both 32 bit and 64 bit. We plan to release POSRA as open source software under the public domain.

References

1. I. V. Filippov, "OSRA: Optical Structure Recognition Application", <http://cactus.nci.nih.gov/osra/>, N.p., n.d. Sept. 12, 2009.
2. D. Weininger. "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules". *J. Chem. Inf. Model*, 1988, 28, 31
3. Noel M O'Boyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch and Geoffrey R Hutchison. "Open Babel: An open chemical toolbox".*J Cheminf*, 2011, 3, 33
4. "Convert, Edit, and Compose Images." *ImageMagick: Convert, Edit, Or Compose Bitmap Images*. N.p., n.d. Web. 02 Apr. 2014.
5. I. V. Filippov, M. C. Nicklaus, "Optical Structure Recognition Software To Recover Chemical Information: OSRA, An Open Source Solution" J. Chem. Inf. Model., 2009, 49, 740
6. T. Halpin, "Object Role Modeling." <www.orm.net> N.p., n.d. Feb. 28 2014.
7. "MySQL :: The World's Most Popular Open Source Database." MySQL :: The World's Most Popular Open Source Database. N.p., n.d. Web. 01 Apr. 2014.
8. D. van Heesch. "Doxygen." <www.doxygen.org>. N.p., n.d. Feb. 24, 2014
9. "Get." OpenMPorg RSS. N.p., n.d. Web. 01 Apr. 2014.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

ActiveDirectory Updated

Ian Blake, Matthew Caruano, Ignacio Llamas, Jeffrey Ma, Paul Scherer

imprivata

Abstract

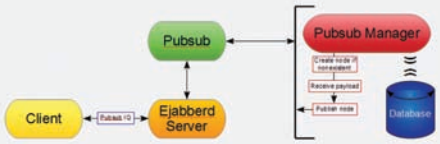
Imprivata Cortext is a secure mobile messaging platform for healthcare professionals. Our objective is to improve upon the existing platform by designing a system to decrease the size of data updates and make changes available for the users on demand.

This roughly translates to two areas of interest:
- Provide incremental data updates
- Convert from a data-pull model (in which the user gets new data only when it is requested) to a data-push model (where the user gets new data as soon as it is available)

Methodologies

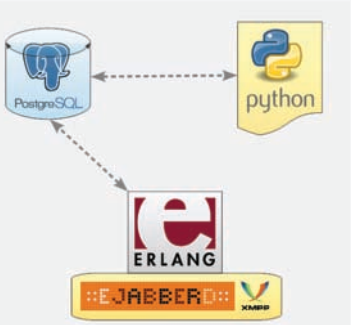
- For the incremental update problem, we create deltas (changes to data), and combine these deltas into a checkpoint system. Only new updates are sent to the clients, as opposed to sending the entire domain directory
  - The checkpoint system monitors and packages deltas accordingly
- To change from a data-pull model to a data-push model, we integrated use of the publish-subscribe ejabberd module.

Publish-Subscribe Implementation



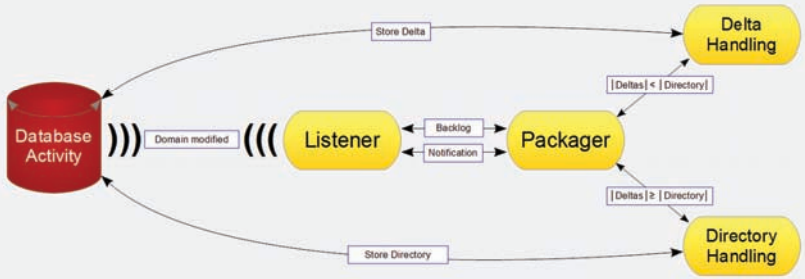
The server compares the timestamp and packages the appropriate delta and/or directory. Subscribed clients will receive the packaged updates as they become available.

Language Interface



The server runs on ejabberd, which is a platform that uses both Erlang and the XMPP protocol. Aside from the server, we use Postgres to help store all the necessary data for the implementation. In addition, there are also Python scripts to help connect the Postgres with the ejabberd server to listen for changes and trigger appropriate actions.

Delta Implementation



The database monitors the clients' updates, inserts, or deletes. There is a server listening for these changes, which packages the updates into deltas or a checkpoint.

Technologies/ Methodologies Used

- **Python** (database interface)
- **PostgreSQL** (database)
- **ejabberd** (XMPP server platform)
  - **Written in Erlang**
  - **XMPP** (messaging protocol)
    - **IQ Stanzas** (client-server queries)
    - **Publish-Subscribe Protocol** (push-model spec)

Results

- Server determines which client update method is more efficient; to send deltas, a checkpoint, or a combination of both
- Clients no longer need to poll the server for updates
- Connected clients are immediately given updates as they occur
- Maintains backwards compatibility

Acknowledgements

Imane Idbihi
Software Engineer, Imprivata

Kevin Carosso
Software Architect, Imprivata

David Kashtan
CTO, Cortext, Imprivata

Devdutt Sheth
Software Engineer, Imprivata

Krutarth Shah
VP, Engineering Mobile Products, Imprivata

Dr. Linda Werner
Faculty Advisor

Capstone Project

Geo-tagged Sensor Fusion

Bardia Keyoumarsi, Michael Bennett, David Tucker,
Vincent Lantaca, Michael Baptist

Abstract

Motivation:

The Lawrence Livermore National Laboratory's Data Science Initiative was established to advance state-of-the-art technology in the "big data" domain including data collection methodology, storage, visualization, analytics, modeling, simulation, and high performance computing capabilities. To support this initiative, the students and faculty of the School of Engineering (SOE) at University of California, Santa Cruz (UCSC) produced a real-time sensor collection system whose acquired data can be fused with publicly available "open" data (e.g. social media, climate data, traffic data, etc.).

Objective:

The goal of this project is to produce a web application that will merge gathered data with publicly available information using standard formats and open platforms. Another dedicated application (for iOS) would allow any iPhone to become an information collection source, and relationships among data can be determined using attached geotags. Related data may be visualized using a number of methods including graphing and geographic plotting. Implementation specifics will need to be clearly documented and cleanly presented.

In addition to the real-time sensor collection system, the GSF project team shall provide tools and frameworks for data query, analysis, and visualization of the fused data for utility and ease of adaptation to later potential applications.

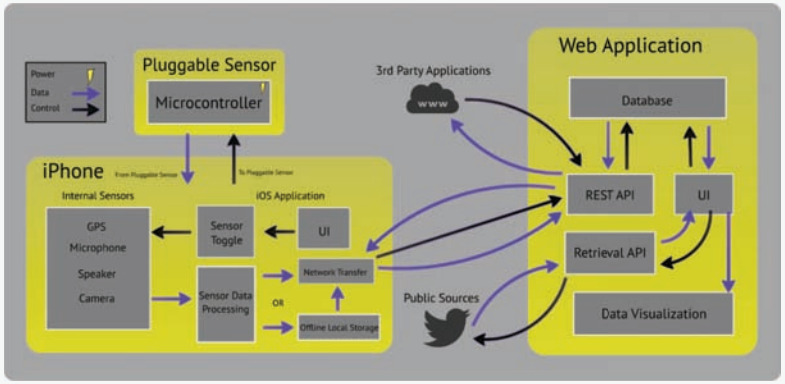
Approach

The Geo-tagged Sensor Fusion (GSF) project uses a dedicated iOS application that interacts with the iPhone's onboard sensors to collect various types of data. In addition, a pluggable sensor module has been designed to interact with the iPhone's headphone jack for additional sensory data. The headphone jack is used for power and communication with the pluggable sensors. Each sensor is identified by our application when plugged in, and transmits data samples to the application.

All the collected data is then sent to a centralized server where it is stored in a NOSQL database. The web application is used to interact with the database to allow users to create their own visualizations. The users have the ability to fuse the collected data with other publicly available "open" data to create meaningful visualizations.

Overview

Design



Software Analysis

Web Application:

The web application suite runs on the Django platform and uses MongoDB as its NoSQL database. The wep app allows users to create visualizations by fusing sensory data collected by the iOS app and tweets retrieved from Twitter. The backend of the web app includes a RESTful API that allows the iOS app to upload data and third party applications to access data from our database. All data is sent in GeoJSON format.

Retriever & Visualizer:

Queries issued on the fusion interface return data contained in our database. Periodically, those queries are sent to Twitter in the background to ensure results remain current and relevant. The retriever responsible for this process filters out Tweets that are not geotagged and packages the rest in GeoJSON format.Data from Twitter is merged with data from the database before being sent to the visualizer which is capable of rendering any valid GeoJSON file. It produces plain visualizations using HTML5 and spatial visualizations using OpenStreetMap and the Leaflet JavaScript API.

iOS Application:

The iOS application is a data collection system that supports iPhone 4 up to iPhone 5s running iOS 7. It allows the user to collect imagery data and runs OpenCV to detect the number of bodies and faces in the image, which is used to estimate the number of people at a GPS coordinate. Other onboard sensors can be turned on via the custom user interface including our pluggable sensor suite. Once finished with a collection session the data is sent to our server securely using custom API keys or saved to be sent later if no network service is detected or whenever the user is done collecting. The system is easily scalable to allow for any type of data to be transmitted.

The application also offers a route planning service which allows an administrator to send a group of GPS coordinates via text message to a field agent. Our service will compute and plot the fastest route to each location so the agent can optimize their time collecting data. They also have the option of getting directions via Safari or launching the route into the native Google Maps application.

Hardware Analysis

Pluggable Sensor:

The pluggable sensor is powered by a sine wave on the left channel of the headphone jack, which is stepped up, rectified, and regulated to around 3V. This is used to power an ATxMega128a4u microcontroller which retrieves data from a digital temperature/humidity via TWI.

Signal Processing:

The communication from our microcontroller to the iPhone is through the mic-line of the iPhone's audio jack. We use a controlled AC signal to transfer encoded data over the communication channel. When the signal reaches the iPhone we process the audio samples and decode the bits. The iPhone can request data from the microcontroller by sending a transmission on the right audio output channel, which is connected to an ADC on the microcontroller.

Results

Over the course of our allotted implementation time, under extreme working conditions, we have developed a mobile sensor suite that pushes sensor population, temperature, and humidity data to our web application. In addition to data collected by the iPhone, we have also developed a retriever that requests data from Twitter. The data is stored in our database and is available for viewing via our visualizer.

A map visualization contains tweets and sensory data found across the world.

GSF iOS Application

Conclusion

The Geotagged Sensor Fusion project will allow users to participate in a global survey that will attempt to merge and visualize the massive amounts of unorganized public data available to netizens today.
The use of sensors allows people to add to publicly available data which could be useful for census-taking, crisis response, and many other applications.

Establishing data relationships provides a better means for producing new research and educating the population as it makes finding relevant information quicker and more convenient.

Acknowledgements

We would like to thank:

- Lawrence Livermore National Laboratory for sponsoring our project and Michael Goldman for his countless hours of feedback and technical support.
- David Munday and Ethan Papp for their guidance, insight and mentorship.
- Terry Figel and Heidi Sitton for giving us access to various ITS services.
- Baskin Engineering Lab Support for their assistance with lab equipments.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Wireless Security Device

Jesus De Haro, Ellis Izumoto, Samantha Tak, Dennis Tran

McAfee

Abstract

Motivation

There are two things that people desire from their passwords and the devices that store them—convenience and security. People have many passwords and user logins that they must remember. It is difficult for a person to come up with different secure passwords for each website and expect to remember them all. Having a device that could identify the user and login to any website he or she desires would add convenience to the user's life. Furthermore, users do not only want convenience, but they also want security. They want the ease of accessing websites knowing that it is unlikely their passwords or online identities will be stolen.

Objective

The goal of this project is to design and implement a wearable device that will link to other devices through wireless communication and have it securely login to websites of the user's choice. The wearable must be secured to ensure that the correct person is using the device. This project will play on the principle of convenience, enabling access to passwords over a range of devices based on proximity.

The project assignment is to create a wearable prototype that will authenticate the intended user of the device. The user is then continually monitored to confirm the device stays on the appropriate person. The prototype will sign into websites that would normally require a password input. The device will do this wirelessly and securely, knowing that it is the proper user without having him or her re-authenticate after the device is initiated. The wearable will also test for forgeries from other users trying to access the wearable device using biometrics.

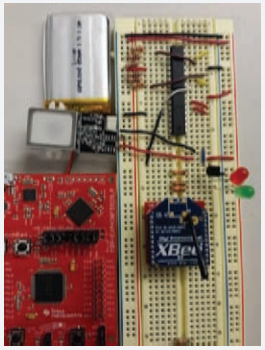
Approach

Our approach to this project was to first research similar products that were currently in the market. This gave us a glimpse into what components and methods we could use for our design. The next thing we had to do was to research individual components and options and check if they fit into our system requirements. The next step was the prototyping stage where we test modules individually such as the fingerprint scanner, skin contact sensor, and microcontroller. After testing each component separately, we integrated them into one system. The final stage was the enhancement stage where we start moving our prototype onto a PCB and cleaning up any errors that would inhibit the user experience.

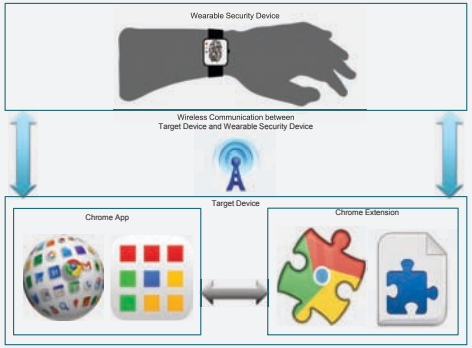
Project Overview

This project's main goal is to create a prototype of a wearable security device that is able to log into websites that require passwords. The device will authenticate the user with biometrics and will continually monitor that the device has not lost contact with the user.

Prototype System



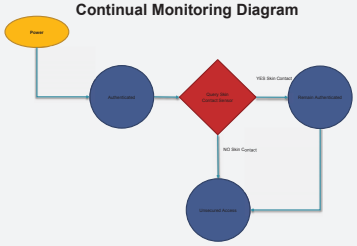
Final Implementation



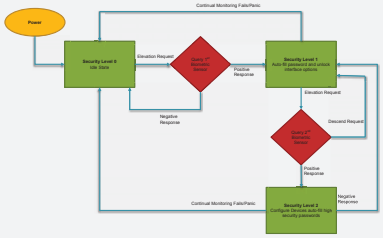
Overall Design

Security Flow Diagrams

Continual Monitoring Diagram



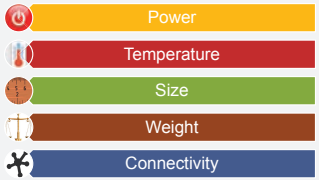
Security Level Diagram



Results & Analysis

System Requirements

- Power: limited to a single rechargeable cell
- Temperature: should be less than 100°F
- Size: area limited to 100mm x 100mm x 20mm
- Weight: must weight less than 8 ounces
- Connectivity: must have two UART interfaces for the fingerprint scanner and wireless communication module



Wearable

The wearable is composed of four main modules: a control unit, a skin contact sensor, a fingerprint scanner, and a wireless communication module. The control unit handles inputs from the other three components and sends commands to both the wireless communication module and the fingerprint scanner depending on the input it receives from the user on the target device.

Target Device

The target device is any device that supports a Google Chrome browser such as a mobile phone or computer. For the design to work properly the user must install a Chrome App and connect a USB wireless communication module.

Overall, our team believes that the project has met the objectives. Our goal was to create a wearable that wirelessly sends data in a secured method and to have the device act as an alternative to filling out and remembering passwords. All this boiled downed to two aspects—security and convenience—and we believe that this project achieved both. In also, within the system requirements we had set for ourselves in the beginning of the project.

The challenging aspects of this project were the unforeseen issues. The implementation was greatly different from what initially planned. For example, we did not expect to implement a Chrome App to fill in passwords. Another example is that we did not expect to fill in passwords for website, we had instead planned to unlock devices instead.

Acknowledgments

Our team would like to first thank the Baskin School of Engineering at University of California, Santa Cruz for giving us the opportunity to participate in the Senior Design course. Also, we thank BELS for providing us with components through the design process. We would like to thank McAfee/ Intel Security, especially Christian Roher, Carl Woodward, and Peter Wyatt. We also want to thank our instructor David Munday and TA Ethan Papp for the guidance they provided.

Capstone Project

Oracle Hardware Compression

Clement Ng ▪ Xikang Pan
Kylan Roberson ▪ Kelly Scanlon

Abstract

In modern distributed systems, many important data processing algorithms are bound not by CPU time, but by power and network bandwidth. Instead of investing more power in a faster interconnect, data can be compressed to improve the apparent bandwidth available to applications. We looked to maximize throughput while minimizing area, power, and delay. We explored this idea by implementing and evaluating a parallel dictionary architecture of the popular Lempel-Ziv-Welch compression algorithm (PDLZW) used in applications such as GNU compress. Our architecture targets TSMC's 40nm ASIC process and has 34 fan-out-of-four (FO4) delays. For a dictionary architecture of 1024 entries, the architecture completes a single cycle of compression in 4 clocks and a cycle of decompression in 3 clocks giving us an average compression rate of 312.5 MB/s and decompression rate of 1.03GB/s. A 300 MB/s link using our architecture could theoretically achieve an average effective bandwidth of 743 MB/s.

Compression

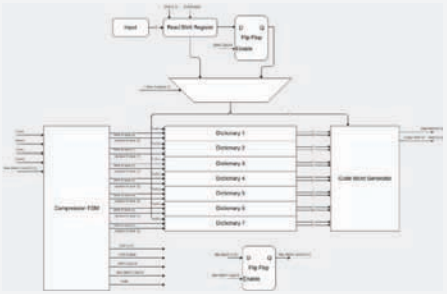


Figure 1: Compression Hardware Block Diagram

A variable byte shift register is used as a buffer to shift in data from some external memory device. The shift register bytes are sent to each CAM which outputs a hit signal and a matching address. The hit and matching address signals are used to generate an output codeword. As shown in figure 2, the codeword reflects the A1 Dictionary format and is calculated to have a bit length of lg(1024)= 10 bits. The A1 format has unique dictionary symbols varying between 2 to 4 bits and varying address lengths. Dictionary symbols with shorter bit lengths have more entries in their dictionaries due to the constraints of a constant codeword bit length

symbol	address
2-4	8-6

Figure 2: Codeword sequence for a 1024 entry dictionary

A valid signal is output on the same cycle so that any device interfacing with the compressor will know when output data should be sampled. The compressor then writes the matched bytes plus the next byte in the buffer to the dictionary and then shifts its input buffer by the number of bytes that were matched. It also signals the shifted amount to an interfacing device so that the device can increment the address counter in memory.

Overview

Algorithm

The PDLZW compression and decompression algorithm splits dictionary entries into multiple dictionary sets of incrementing byte denominated sizes. In compression, these dictionaries are implemented so that can be searched in parallel. When a match is found, the compressor will output a codeword which is given as the concatenation of a unique identifying bit pattern representing the matching dictionary and the address within the dictionary that matched. The total number of dictionary entries determines the length of each codeword, which is determined by lg(n), where n denotes the total amount of dictionary entries: a 1024 entry dictionary space will have a codeword length of 10 bits. The bits not needed to represent the address space are therefore used to represent the dictionary symbol. Dictionary 0 which is always a 256 entry dictionary represents the 256 characters of the ASCII table for both compression and decompression algorithms. The dictionaries PDLZW decompression algorithm have an asynchronous read and a synchronous write. The decompression algorithm takes in codewords and reconstructs the dictionaries created in the compression algorithm.

PDLZW Example

Figure 3 represents compression on an input string, aaabbccbbccaaa. The algorithm shifts in a variable amount of bytes and reads a certain amount of bytes based on the max matched dictionary number. At first, the max match is initialized to 0. We search for "a" and find that the longest match is found in dictionary 0. "a" is part of the first output codeword. Therefore Our concatenated string, "aa", is stored into the next dictionary, which is dictionary 1. Then we shift in new characters into the buffer and repeat the process over again. We find the longest match, "aa", exists in dictionary 1. So we output "aa" and store the new concatenated string "aab" into dictionary 2.

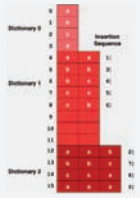


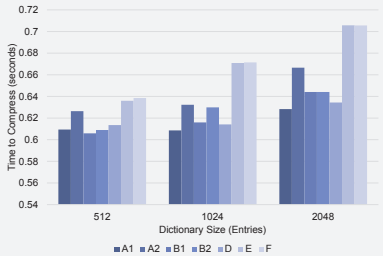
Figure 3: PDLZW Compression Example

Compression

The PDLZW compression takes in a sequence of characters by shifting in a variable amount of bytes into a buffer. Characters are stored in multiple dictionaries of incrementing byte lengths. Dictionaries are represented by content addressable memory (CAM) in hardware. The algorithm detects if the sequence of characters from the input buffer have appeared in any of the dictionaries in parallel, finds the longest match, and outputs the codeword. Codewords are a concatenation of two variables, the max matched dictionary number and the matched address from the CAM.

Decompression

The PDLZW decompression takes in compressed codewords and recreates the dictionaries from the PDLZW compressor. Unlike in PDLZW compression, the dictionaries in PDLZW decompression are represented by SRAM cells. The decompressor detects the new codeword and saves the previous codeword. The algorithm then concatenates the decompressed substring of the previous codeword and the first character of the decompressed substring of the new codeword and writes the concatenated string into the corresponding dictionaries.



Graph 1: Compression Time vs. Dictionary Size Variations and Formats

	Compression	Decompression
Cycles per codeword	4 clocks, 4ns	3 clocks, 3ns
Power	175mW	168.426 mW
Area	0.226 mm²	0.173 mm²
Energy per bit	70 pJ/b	50.5 pJ/b

Table 1: Hardware Performance Metrics for Dictionary Size 1024 and format A1

	Compression	Decompression
Time per codeword	1.23 us	0.152 us
Power	1.15W	0.65W
Energy per bit	141.5 nJ/b	9.9 nJ/b

Table 2: Software Performance Metrics for Dictionary Size 1024 and format A1

Results

From Graph 1, the compression time of the 512 entry dictionary is significantly improved compare to the larger dictionary sizes. Our design is based on the 1024 entry, A1 format. This format uses eight dictionaries of size 256, 256, 128, 128, 64,64, 64, and 64 entries for dictionary 0 to 7 respectively. To optimize our design, we would use 512 bytes spread over eight dictionaries to save area which then decreases power. This will also decrease the time it takes to compress the data. The format and size we have now has one of the fastest time to compress, but uses more power and area then a dictionary size of 512.

By comparing the performance metrics for hardware in Table 1 and the performance metrics for software in Table 2, we can see that the time to compress and energy per output bit of our design were 3 orders of magnitude better than the software implementation. The design is capable of producing an effective bandwidth of 320 MBps for a 125 MBps interconnect (i.e. 1 GigE) and is capable of improving the effective bandwidth for interconnects up to 750 MBps. With a standard 500 mW ethernet controller we could reduce the energy required per bit by 160pJ an approximate energy savings of 34%.

Decompression

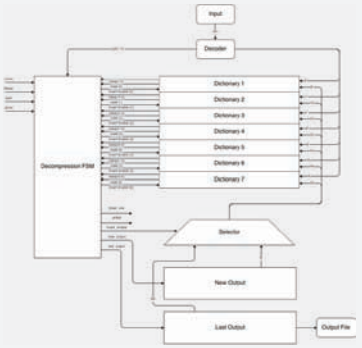


Figure 4: Decompression Hardware Block Diagram

The design takes in an input of compressed codewords which contain lg(n) bits, where n denotes the total number of entries in the dictionary set. This codeword then goes into a decoder, which outputs the dictionary number based on the unique identifying bit pattern representing the matching dictionary and the matched address in the matching dictionary, which vary in bit length. The matched address points to SRAM cells which represent the various dictionaries. The dictionary number from the decoder then points to a state machine. The state machine is responsible for outputting decompressed strings to the decompressed file, outputting the previous and current decompressed strings from the codewords, and outputting an enable signal which is responsible notifying which dictionary to write data into. The previous and current decompressed strings, which are the last output and new output respectively, is sent to a selector. The selector is responsible for concatenating the last output with the first character of the new output which is then sent to the dictionary which will be updated based on enable signal from the state machine. This process is responsible for reconstructing the dictionaries used in PDLZW compression.

Acknowledgements

The Oracle Hardware Team wishes to thank the following for their support:

Hesam Moghadam
Oracle Corporation

Nathan Pemberton
Oracle Corporation

David Munday
UCSC

Pat Mantey
UCSC

Vikas Aggarwal
Oracle Corporation

Jose Renau
UCSC

Ethan Papp
UCSC

References

[1] M.-B. Lin, "A parallel VLSI architecture for the LZW data compression algorithm," J. VLSI Signal Process., vol. 26, no. 3 pp. 369-381, Nov. 2000.
[2] M.-B. Lin, J.-F. Lee, and G. E. Jan, "A lossless data compression and decompression algorithm and its hardware architecture," *IEEE Trans. Very Large Scale Integration (VLSI) Syst.*, vol. 14, no. 9, pp. 925-936, Sep. 2006.
[3] M.-B. Lin, Y.-Y. Chang, "A New Architecture of a Two-Stage Lossless Data Compression and Decompression Algorithm", *IEEE Trans. On Very Large Scale Integration (VLSI)*., vol. 17, no. 9, pp. 1297-1303 Sep. 2009

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Software Compression Analysis

Edward Kuang ▪ Christopher Selling
Eric Yen ▪ Nicholas Brower

ORACLE

LZ4

Abstract

In modern distributed systems, data processing algorithms are bound not by CPU time, but by power and network bandwidth. The Oracle Software Team has evaluated algorithms on a power-efficient ARM architecture. LZ4 is a competitive compression algorithm recognized for its fast compression speed. The Oracle software team has multi-threaded LZ4 and analyzed the algorithm for optimal thread count, block size, and dictionary size.

Approach

LZ4 is a lossless compression algorithm based on the LZ family of algorithms. The original algorithm splits a file into a large array of blocks and then compresses each block independently. Compression is achieved by matching a minimum of 4-bytes to previous bytes seen. Matching is achieved by storing pointers to 4-bytes inside a hash-table and comparing against those bytes to determine if a match has occurred. If a match has occurred, record the length of the match and the location of the match in the hash-table.

The first step was to analyze LZ4 and determine the algorithm's efficiency and competitiveness. While the original algorithm is very competitive, it did not make use of all available processor cores. The multi-threaded approach we came up with was a dictionary that corresponded to the order that blocks were read in (Figure 3). The next step was to benchmark on the ODROID. LZ4 was analyzed with respect to thread count, dictionary size, and block size. In the end, an analytical model was constructed to highlight bottlenecks in hardware that could potentially improve the efficiency of the algorithm.

Analysis

From the Thread Usage vs. Effective Bandwidth graph we can see that there is a linear relationship between thread usage and effective bandwidth from 1 to 4 threads. This is because the ODROID's CPU has 4 cores, and spawning 4 threads allows the kernel to utilize the cores efficiently. The effective bandwidth versus the number of cores used can be described by the function f(x)=116.5x+34.375. This means that the more cores we add to the CPU, the linear increase in effective bandwidth will likely continue. However, we do see that at the rate of increase in effective bandwidth flattens out a little; therefore, this suggest that there may be a point of diminishing return.

Results

The corpus provided to us by Oracle comprised of 4 files that are very similar to the format of TPC-H benchmarks. The average effective bandwidth of LZ4 (the rate at which files are compressed) is 489 MB/s. This was accomplished using 4 threads, a hash-table of 512 bytes, and 8 KB blocks. The LZ4 compression algorithm on average reduced the files size in the corpus by 30%.

It can also be seen in Figure 2 that the amount of processor cores show a linear increase in effective bandwidth. Because the ODROID has 4 cores, the results show that 1 thread for each core is optimal.

Conclusion

In regards to Oracle's investigation in a hardware/software evaluation of compression algorithms, LZ4 is moderately comparable to the implementation of LZW by the hardware team. While dedicated ASIC's would provide the highest effective bandwidth, the advantages of a slightly slower software implementation is still beneficial. The LZ4 code is constantly being improved upon and can also easily be ported onto different devices.

Overview

LZ4

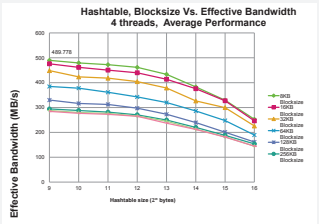


Figure 1. The average effective bandwidth of the LZ4 compression algorithm for each hash-table and corresponding block size. For the same block size, it is evident that a larger hash-table reduces effective bandwidth.

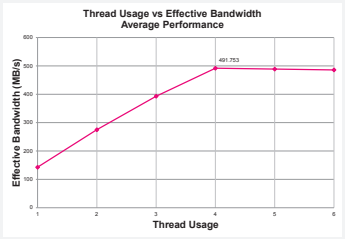


Figure 2. With one thread per core, it is shown that there is a linear relationship between effective bandwidth and threads.

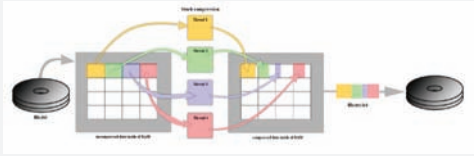


Figure 3. The multi-threading approach that LZ4 uses. A file is read from disk into a buffer in memory. The file is then compressed and written to an output buffer in memory. The output buffer is written to disk. This was necessary to emulate the behavior of hardware compression.

Figure 6. ODROID-U3 with specs.

ODROID Specs

- Exynos 412, 1.7 GHz, quad-core processor.
- ARM Cortex-A9 architecture
- 2GB DDR2 Ram
- 32 KB L1 cache
- 1 MB L2 cache
- Listed memory bandwidth: 6.4GB/s
- Linux XUbuntu 13.10
- 5V, 2A Power

LZWPD

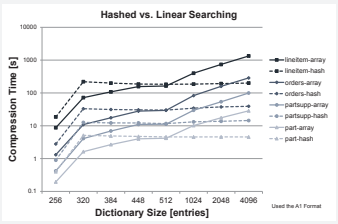


Figure 4. Two different dictionary search methods for LZWPD are compared (Hashing, and Linear Search).

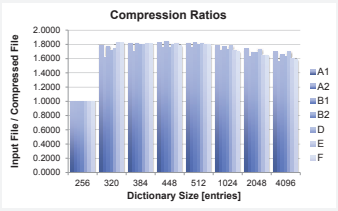


Figure 5. Average compression ratios are compared for all dictionary sizes and formats.

Acknowledgments

The Oracle Software Compression Analysis team wishes to thank the following for their support:

Nathan Pemberton, Oracle Corporation
Vikas Aggarwal, Oracle Corporation
Hesam Fathi, Oracle Corporation
Pat Mantey, UCSC
David Munday, UCSC
Ethan Papp, UCSC
Jose Renau, UCSC

LZWPD

Abstract

LZWPD is a highly parallelizable compression algorithm that is optimal for hardware implementations. The software team has implemented a compatible serial version of this algorithm to compare with results gathered from the hardware implementation. In addition to functional equivalence, the software team has also analyzed optimal dictionary sizes, formats and search methods.

Approach

The software implementation of LZWPD was based on the hardware team's high level block diagram. Program execution follows the following steps: read in a series of bytes, search through the dictionary for matches, and encode the address of the entry where the match was found. While the software implementation could not match the O(1) dictionary search time of the hardware implementation, the software implementation gives preference to matches found in the higher order byte-storing dictionaries. The first implementation performed linear search in order to verify that output was comparable with the hardware team, but use of profiling tools showed 95% of runtime was allocated to this search time. The next implementation used hashing with collision resolution into binary trees to reduce to this search time. In this implementation circular buffers were used to ensure correct ejection from the hash-table to maintain compatibility with the hardware implementation. Finally, new dictionary formats and sizes were implemented and measured for optimal compression ratios.

Analysis

When converting from linear search to hashing, it became obvious that no series of optimizations would make LZWPD a compatible algorithm for software, especially in the context of real-time applications, the compression times were simply too high. At this point the software side of LZWPD became focused on finding an optimal dictionary size and format for use by the hardware team. As seen in Figure 6, all dictionary sizes and formats were measured for compression ratio. As hardware compression is extremely fast, compression ratio became the primary interest. One interesting result to note is that increasing the dictionary size past 512 clearly decreases compression ratio due to longer encodings.

Results

Reviewing the results seen in Figure 5, runtime improvements were obtained through changing the search method of LZWPD's dictionary from linear search to hashing. Keeping in mind the logarithmic scale of the y-axis and exponential scale of the x-axis, all linear search results are O(n) + O(m) where n is proportional to dictionary size, and m is proportional to file size. Similarly, all hashing results are O(m).

In Figure 6, we can see an optimal average compression ratio is found for a dictionary of size 448 with format B1. The optimal average compression ratio is found to be 1.8423 and calculated as an average across all four input files. The maximum compression ratio observed was found to be 2.1415 when run on the "lineitem" file with a dictionary size of 448 and format B1.

Conclusion

In conclusion, LZWPD does not seem to have a place in software due to its very low compression speed. There is still room for improvement if compatibility with a hardware implementation is not desired, but such optimizations are predicted to still fall short of other software implementations such as LZ4. Comparisons to the hardware implementation display three orders of magnitude worse compression time and energy consumption when implemented in software and run on the ODROID.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Variable Signal Delay

John Gustafson and Kelly McNutt

Raytheon
Applied Signal Technology, Inc.

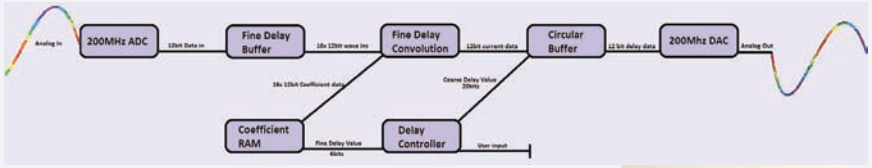
Abstract

Applied Signal Technology has funded this senior design project to create a variable signal delay. A signal delay is something that can increase the phase of a wave without distorting the signal. The goal is to create a system that takes in an analog signal and output the signal with a delay. The delay can be as small as 50 picoseconds and as much as 40 microseconds. To implement this delay we will be using a convolution on the input wave with a shifting set of coefficients.

Approach

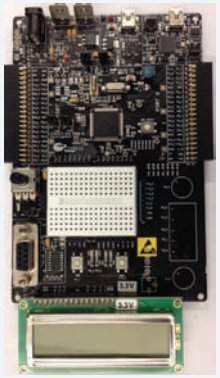
- Main Components
  - The system is made up of a few parts: The delay controller, the fine delay, the course delay, the DAC, and the ADC
  - A user interface is needed to specify the time delay
  - The prototype is made on the Psoc5 Cypress Controller
  - The final design is made on the Xilinx Kintex 7 FPGA
  - The DAC and ADC are implemented on the FMC150
- Fine Delay
  - Takes in data from the ADC to a 16 by 12 bit buffer
  - Uses the buffer to do a 16 tap convolution with sinc function coefficients
  - To create a delay, a different set of 16 equally spaced taps are chosen from the sinc function
  - The set of coefficients used are determined by the delay controller
  - The output is sent to the course delay
- Course Delay
  - Takes in data from the fine delay
  - Uses a circular buffer of length 1048576 by 12 bits to store the wave
  - The delay controller tells the course delay what address to output to DAC
- Delay Controller
  - The human interface with the fine and course delay
  - The delay controller evenly changes the fine delay

Overview



- The delay controller acts as the brain of the microcontroller. It controls the delay values that alter how much the rest of the design delays the output.
- User input drives the delay controller, which then drives the circular buffer and the coefficient RAM. The fine delay convolution grabs data from the coefficient RAM. The rest of the process runs ADC to DAC.

Top: FMC150 Daughter Card



Left: Cypress Psoc5 Microcontroller

Right: Xilinx KC705 Microcontroller

Acknowledgments

- Michael Ready – Sponsored Advisor
- John Vesecky
- Jeff Bertalotto
- Anujan Varma

Analysis

- There are two main obstacles to overcome when designing a digital filter.
  - The math behind the fine delay
  - Parallel programing in VHDL
- The fine delay works by having 1024 coefficients hardcoded into memory. The filter grabs 16 equally spaced taps and pipeline multiplies these taps with the ADC 16 by 12 bit buffer. The 16 numbers are then cascade added and output to the circular buffer.
  - Shifting the coefficients causes a delay by centering the 16 incoming wave data points around the center of the set of 16 coefficient taps chosen.
  - Each operation had to be done in sets of one clock cycle to keep the speed around 200MHz. Each function had to be pipelined and done in increments.
- The course delay is a circular buffer in VHDL which utilizes its own programed RAM.

Results

The Psoc5 prototype has a working fine delay and course delay but the code runs slowly. The fine delay on the Psoc5 has a amplitude distortion when the fine delay shifts, but does increase the delay precision up to 64x. The KC705 final design runs a 245.76 MHz sampling rate and gives us 2.95Gb/s of data to process.

Conclusion

The coefficient shifts are able to increase the resolution of a phase delay as long as the code is properly implemented in parallel. Otherwise the added code slows down the throughput to a point where implementing a pure coarse delay would have more precision. At our optimum we can increase the resolution of our sampling frequency by 64x.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Serene Source

Gahl Levy, Michael Parker, Daniel Shubin,
James Thomas, Aastha Verma, Sable Yemane

SERENA™

Chef Plugin

Abstract

The Chef Plugin allows users of Serena Release Automation to interface with Chef, an application management tool. Chef describes infrastructure as series of resources that should be in a particular state, packages that should be installed, services that should be running, and files that should be created. This plugin allows users to easily create and automate Chef tasks which manage Chef resources and provision and configure machines.

Technologies Used

- **Serena Release Automation** – A platform to automate the deployment of new software and updates to machines both locally and in remote locations.
- **Chef** – A tool for configuring and maintaining servers as well as provisioning and configuring new machines.
- **Groovy** – Cross Platform Scripting language used to execute the processes in the plugin in Serena Release Automation.
- **XML** – A markup language used for User Interface definitions of plugins for Serena Release Automation.
- **Maven** – Build automation tool that describes how software is built and its dependencies. Used to build the plugin for the Serena Release Automation Software.

Approach

- Install, configure, and learn how to use Chef.
- Determine necessary functionality for Chef Plugin.
- Write Groovy scripts to execute plugin tasks.
- Define plugin user interface with XML.
- Build Groovy scripts and XML into a plugin for Serena Release Automation with Maven.

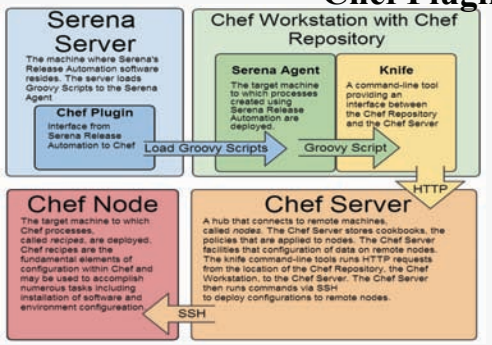
Results

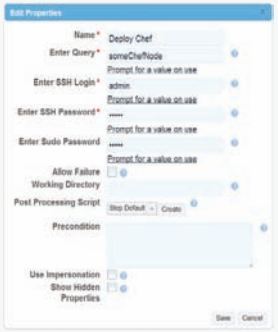
- The plugin allows for the automation of Chef processes through Serena Release Automation
  - Automated the deployment of chef recipes.
  - Automated management of assets on chef server

Motivation

DevOps bridges the gap between software development, IT, and QA to make software release and management faster, easier, and more consistent. This is accomplished partly through release automation: the automated deployment of software, updates, and machine configuration. Continuing in this theme, the goal of this project is to abstract away details of release automation to make it a simpler process for DevOps engineers. This was done by creating new functionality for Serena Release Automation: a Chef plugin and a Mobile application.

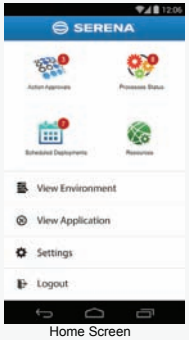
Chef Plugin



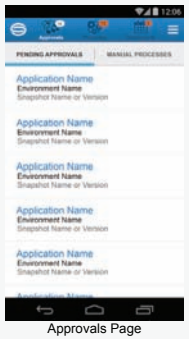


Mobile Application

Login Page



Home Screen



Approvals Page

Conclusion

By creating a plugin and a mobile application for Serena Release Automation, this project helps to expand the scope of Release Automation by allowing users to automate Chef processes and have faster and simpler access to Serena Release Automation. The Chef plugin allows DevOps engineers to maintain their existing Chef infrastructure while gaining access to more powerful features provided by Serena Release Automation. The mobile application for Serena Release Automation gives DevOps engineers the flexibility to interact with Serena Release Automation from anywhere.

Serena Mobile

Abstract

Serena Mobile gives DevOps engineers the ability to manage Serena Release Automation from anywhere. The mobile application allows for the approval of process requests, the requesting of new processes, and viewing the status of processes that are currently running or scheduled for the future. These features allow for the deployment of software, updates, and machine configuration without the extra navigation that is needed when using a web browser.

Technologies Used

- **Serena Release Automation** – A platform to automate the deployment of new software and updates to machines both locally and in remote locations.
- **FluidUI** – Tool to create mockups for mobile user interfaces.

Approach

- Determine which core features of Serena Release Automation that would give the most utility to a mobile application.
- Create a mockup user interface to test the flow and the layout of the mobile application.

Results

- Developed a functioning mobile application
- Mobile features included:
- Approval of Process – allows users to approve or deny processes.
- View Process- allows users to view running or scheduled processes while also being able to pause or cancel a process.
- Request Process- allows users to schedule new processes.

Acknowledgements

- Faculty Advisor: Dr. Linda Werner
- Corporate Mentor: Julian Fish, Director of Software Development at Serena Software
- Corporate Mentor: Ashish Soni, Senior Software Engineering at Serena Software

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Interactive Visualization of the Next Generation Science Standards

Austin Coleman, Stephen Domenici, Jonathan Eboh,
Mesuilame Mataitoga, Henry Ta

tanzle

Abstract

Sponsored by Tanzle, Inc., our application is an Interactive Visualization of the Next Generation Science Standards (NGSS). “The Next Generation Science Standards establish learning goals in science that will give all [K-12] students the skills and knowledge they need to be informed citizens, college ready, and prepared for careers1.” Our application allows the display and manipulation of a 3-dimensional representation of the NGSS. This gives teachers and curriculum designers an improved experience when viewing the NGSS on zSpace2 iPad, Android, and web platforms.. zSpace is an immersive, interactive 3D display.The zSpace version of the application allows users a level of interaction that is not present in existing textual representations.

Our internal 3D representation of the NGSS will allow teachers and administrators to use the 3D capabilities of the zSpace computer to visualize this graph. Users can manipulate the graph along all three axes. The iPad allows users to manipulate a 3D representation of the NGSS.

Usability

Our Visualization of the NGSS is optimized for human intuition via the use of cognitive levers. It is targeted toward teachers and curriculum designers that are exploring or have adopted the NGSS. We’ve designed the tool to cut through the complexity of the NGSS. Visualization of the NGSS focuses on visualizing and connecting the different performance expectations of the NGSS that would otherwise be difficult in a strictly text based document format.

Our visualization offers two methods of interaction. The first method is accessible through the zSpace hardware, which uses a combination of trackable eyewear and a trackable stylus to produce an image that has depth relative to the user’s surrounding. The second method is offered through the iPad hardware which displays an image that has depth relative to the screen display on the iPad. This display on the iPad uses the touch interface to interact with the graph.

Technologies Used

Back-end
- JSON for parsing the data.
- Unity C# to build an internal representation of the NGSS data.

Front-end
- NGUI for user interface.
- zSpace virtual holographic display, stylus, glasses
- iPad, Finger Gestures for generating meaningful interaction.

Figure 1: 3D Display

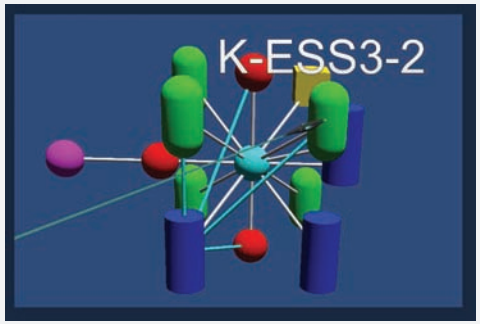
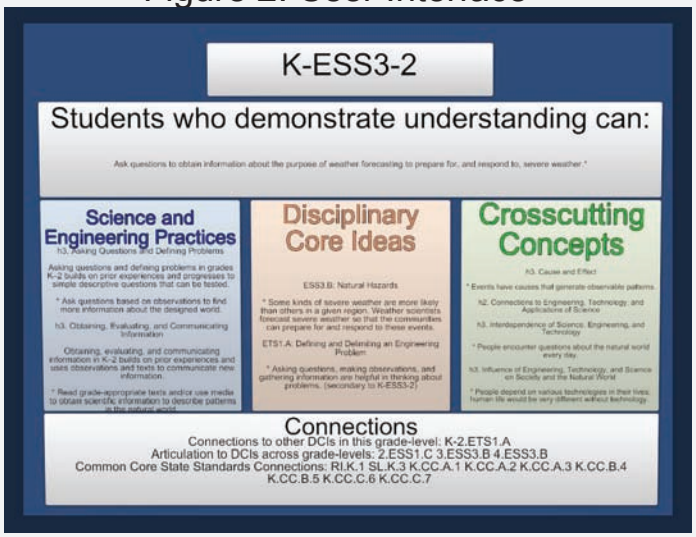


Figure 2: User Interface



Acknowledgments

Nancy Clemens, Founder and CEO, Tanzle, Inc.
Dr. Julian Gómez, VP of Development, Tanzle, Inc.
Mike Vesely, Co-founder and President, Tanzle, Inc.
Dr. Linda Werner, Faculty Advisor, UCSC
Oliver Davies, software engineer, Tanzle, Inc.

Overview

The NGSS is described by their website as a set of national education standards that are “rich in content and practice, arranged in a coherent manner across disciplines and grades to provide all students an internationally benchmarked science education.” Visualization of the NGSS separates the standards into nodes. Each node corresponds to a performance expectation for a student at some grade level or grouping of grade levels. The performance expectations are comprised of three dimensions: Science and Engineering Practices (SEPs), Disciplinary Core Ideas (DCIs), and Crosscutting Concepts (CCs). The SEPs “describe behaviors that scientists engage in as they investigate and build models and theories about the natural world and the key set of engineering practices that engineers use as they design and build models and systems. The DCIs focus on the “K-12 science curriculum instruction and assessments on the most important aspects of science.” The DCIs are divided into four domains: the physical sciences, the life sciences; the earth and space sciences; and engineering, technology, and the applications of science. The CCs tie together the broad diversity of science and engineering core ideas. They provide students with connections and tools related to different disciplinary content.

For example, Fig. 2 displays different pieces of the performance expectation named “K-ESS3-2,” which refers to a performance expectation in the DCI of Earth and Space Science that deals specifically with Earth and human activity. Fig. 2 shows how students can demonstrate understanding of K-ESS3-2. The SPEs for K-ESS3-2 include “Asking Questions and Defining Problems” and “Obtaining, Evaluating, and Communicating Information.” The associated DCIs are “Natural Hazards,” of the earth and space sciences DCI, and “Defining and Delimiting an Engineering Problem,” of the engineering, technology, and the applications of science DCI. One of the CCs includes “Cause and Effect,” where “Events have causes that generate observable patterns.”

The third section of Fig. 2 is comprised of the three types of connections: In-grade connections, which are connections to performance expectations within the same grade; Across-grade Connections, which are connections to performance expectations within other grades; and Common Core State Standard (CCSS) connections, which are connections to information presented in the CCSS. There are over 1000 connections between the performance expectations. Our 3D representation of the connections gives a better understanding of the connections than a text-based representation of the connections. Fig. 1 shows A 3D visualization of the connections where the different node shapes and colors correspond to different DCIs associated with the corresponding performance expectations.

We are very pleased to include posters for the Senior Design Projects that were done without industry sponsors. Some of these projects were instigated and/or sponsored by research at the Baskin School (e.g. CITRIS, CenSEPS), while others were created by students with the assistance of faculty mentors and TAs.

We have selected three of these projects for presentation in the program, and all were invited to display their posters that summarize their projects.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Motorized EMG-Controlled Hand

Taylor Furtado, Kyle Lawrence, Alexander Lynchosky,
Ivan Romero, Michael Sit

CITRIS

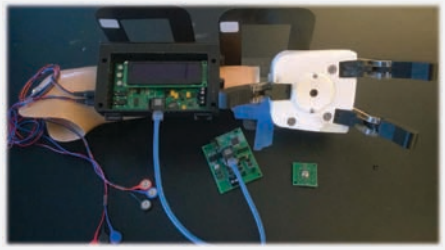
Motivation

This project was proposed by Taylor Furtado, a congenital amputee with over a decade of experience using traditional prosthetics.

Although the controls of traditional hook and pulley prosthetics are simple to learn, these prosthetics can cause significant discomfort after prolonged use, and lack dexterity. The total cost of maintenance and production of such prosthetics can reach upwards of $30,000. Alternatively, modern myoelectric prosthetics provide higher functionality, however these may cost the user over $100,000.

We sought to develop an inexpensive prosthetic that would be capable of grasping and rotating a variety of objects. This prosthetic gives the user a higher degree of dexterity, while providing feedback from it's operating environment. The components we chose for our design have led to a functioning prosthetic that costs less than $1,000.

Approach



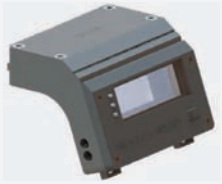
A key concept of our design was product accessibility. We actively sought to purchase materials that were both inexpensive and easy to use for the experienced hobbyist. To reduce the cost of production, we used a 3D printer to create the mechanical structure of the prosthetic.

We also wanted our prosthetic to have a higher degree of functionality than traditional prosthetics, to do so we developed a means to articulate fingers individually and integrate a rotating wrist.

Most importantly, we wanted the prosthetic to be comfortable. Rather than actuating the prosthetic via a user strenuous pulley, we used sensors that detected surface muscle activity in the shoulder to drive the hand. We added features that enables the user to calibrate the feedback responses of the prosthetic, so that they can tailor the device to their needs.

Overview

The hand component performs all of the actuation as indicated by the user and collects environmental data to be sent to the commander.



The prosthetic is capable of opening and closing three individually actuated fingers. To increase finesse, a mechanism was devised to allow the fingers to curl around objects. A wrist was designed that could rotate grasped objects 180º.

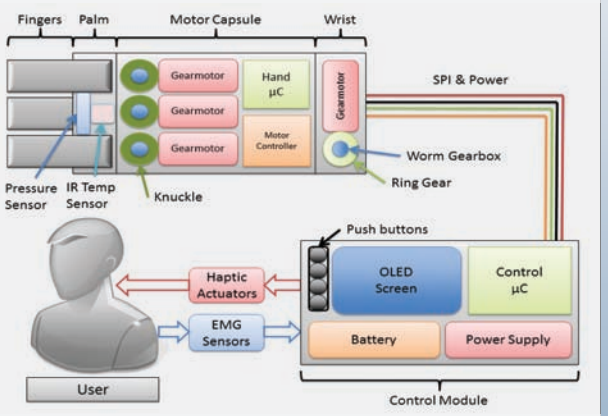
User commands are monitored by ElectroMyoGraphic sensors (EMGs). The EMGs can measure the surface activity of one group of muscles to open or close the fingers accordingly. The second set can monitor for wrist commands. A single flex can indicate left spin, and two sequential flexes will cause the hand to spin right.

The command module receives sensor data from the hand in order to monitor for potentially damaging environments. This information provides the user with tactile and visual feedback, alerting the user of potential risks.

The design of our hand enables it to be mounted onto most standard prosthetic bases. The fingers are constructed to be easily replaced or altered, giving the user the opportunity to make their own modifications.

The design allows the user to modify the responses of the prosthetic. The user may configure the prosthetic through the graphic user interface (GUI). Alterations can disable sensor responses or adjust the level of sensitivity.

Modifications allow the user the opportunity to alter the actuation of the hand. These can be used to lock a finger in place or disable features. Through a series of intuitive button presses, the user can make the prosthetic conform to their needs.



Acknowledgments

On behalf of the M.E.C.H. team, we would like to extend our gratitude to Paul Naud, Professor Mircea Teodeorecu, Professor Gabriel Elkaim, Max Dunne, and Professor Jacob Rosen for their guidance.

We would also like to thank CITRIS, Crown College, and Baskin School of Engineering for providing funding and support.

Analysis

Our developments have led to the creation of a prosthetic that can be successfully controlled by user commands. The hand is capable of grasping and curling around objects, and in addition, the rotating wrist provides a higher degree of freedom unseen in many typical prosthetics.

We were successful in creating this device, while maintaining a budget of $1,000. By providing a GUI, we also allowed the user a means of adjusting the prosthetic to their needs.

Results

The following image is a completed model of our prosthetic. Included in the image are the EMG sensors used to read user controls, the command module and OLED display, as well as the prosthetic hand.

Conclusion

Upon integration our model was capable of grasping and lifting objects weighing up to five pounds. Through extensive searches, we found inexpensive components, the final cost of our model was $916 including estimated tax and shipping. This successful prototype is indicative of future models that have the potential of offering a higher degree of control at a lower expense. Overall, we were successful in achieving the desired functionality for the prosthetic, and were able to expand upon our model to increase it's overall usability.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Motorcycle Safety System

Olivia Krzeminski, Zach Nissen, Yu Chung,
Matthew Stormo, Andrew Bao

Raspberry Pi

We use a Raspberry Pi small form factor PC to process video through a 5 MP camera module. OpenCV and UV4L are used to read the video and find vehicles. If they behave dangerously, the Raspberry Pi sends an alert to one of the UNOs. Each component is explained more below:

- Camera Module
  - Takes 1080p video at 30fps for detecting cars
- Raspberry Pi
  - Runs OpenCV and feedback software
  - Communicates over SPI and UART with the UNOs
- OpenCV
  - Computer Vision library to process video data
  - Detects cars in the video each frame
- Local Binary Pattern Cascade
  - Before the detection phase, this xml file is Generated with 10,000+ images on a server to speed up detection.
  - Demonstrates to OpenCV what a car should look like.

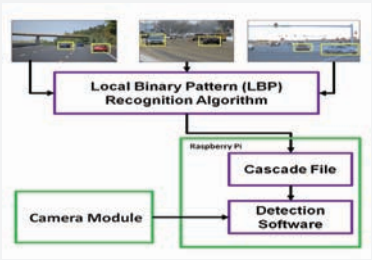


Figure 1: Local Binary Cascade Diagram

Approach

The system structure of our design can be seen below in **Figure 2.** The key component of our system is two Uno32 microcontrollers, one on the motorcycle and one on the feedback vest that controls input from the peripheral sensors and apply PWM feedback to the rider. The other essential component is the Raspberry Pi microcontroller which analyzes all cameras input and handles all images processing for the project. The two microcontrollers communicate via SPI where the Raspberry Pi serves as the Master and the Uno32 the slave.

Figure 2: System Block Diagram

Overview

Figure 3: Sensor Locations

Sensors

Figure 4: Microwave Sensor with Antenna

HB100 Microwave Sensor

- Use to detect danger in front and behind the motorcycle by measuring both speed and distance of approaching objects.
- Capable of accurately detecting vehicles up to 10 meters away.
- Our system uses these sensors to primarily make up for the range under 10 meters that the camera module is incapable of seeing.
- Mainly used for low speed detection; up to 15 mph differential between the motorcycle and the object of interest.
- Incorporates a 1/8" steel horn antenna that reduces the detection angle to 23 degree. This reduces detection outside of the frame of the road.

Figure 6: Raspberry Pi Camera Module

Raspberry Pi Camera Module

- 5 megapixel camera.
- Delivers 1080p video to the Raspberry Pi at approximately 30fps.

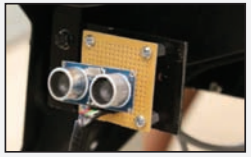


Figure 5: Ultrasonic Sensor

HC-SR04 Ultrasonic Sensor

- Can accurately measure distance up to 4 meters away.
- Primarily used to detect incoming danger to the side of the motorcycle; senses when a vehicle is merging into a motorcycle's lane.
- Our design uses two on each side of the motorcycle. We cross reference data received from both to determine when the motorcyclist is lane splitting. This reduces the number of false positive alarms that our system triggers.

ChipKit Uno

The chipKit UNO32 microcontroller used in this project has two main functions. One UNO controls the feedback system to the driver located on the vest through wireless communication. Second, two bike UNOs control all sensor input and data handling excluding the camera modules and pass on relevant data to the vest. A few of the key features of the UNO are highlighted below:

•SPI
- Communication between UNOs.
- Communication between an UNO and Raspberry PI.
- Reading the velocity of the bike.

•UART
- Communication to and from XBEE allowing wireless communication.
- Communication to computer for testing.

•Change Notice interrupts
- Detecting distances for ultrasonic sensors.
- Detecting relative speeds for vehicles behind and in front of motorcycle.

•ADC
- Reads analog from microwave sensor to detect relative distance.

Haptic Feedback

As vehicles are detected by the sensors in our system and we process a certain detection to be dangerous to the rider, we will send a response directly to the rider. We implemented a haptic response in the form of vibrational motors that are mounted at four different locations on a motorcycle vest. The four locations of the vest are chosen in such a way that it will tell the rider specifically where there a dangerous vehicle was detected. The motor location are on the left and right back shoulder blades, and front left and right breast areas; each location will be trigger when a danger is detected at that respective direction on the road.

Analysis

One of the goals of our project is to detect danger from 0 up to 60 mph. We can figure out the braking distance relative to the speed of the motorcycle from the equation:

$$S = V_0(t_r + t_b) + \frac{1}{2}at_b^2.$$

Where s is distance, V_0 is the speed of the motorcycle, t_r is reaction time, a is the deceleration force, and t_b is braking time.

The average person's reaction time is roughly 0.6 seconds. We also make the assumption that a is approximately 0.7G, which is relatively standard for most motorcycles. Using this, as well as other information gathered from the HB100, we can solve this for the braking time.

For every traffic case, we assume the worst case scenario when determining when to alert the rider. For example when calculating the minimum distances, we use the assumption that the asphalt is wet. This allows our system to handle a wider variety of situations that riders might encounter.

We found that it takes a motorcycle 66.7 meters to stop completely when traveling at 60 mph. This is after the rider has reacted; the 66.7 meters is solely the distance it takes the motorcycle to stop. From this calculation, we found that our system needs to be able to detect vehicles up to 70 meters away with a fast enough response time to still be able to alert the rider before they exceed the minimum stopping distance.

UC SANTA CRUZ
Baskin Engineering

Capstone Project

Portable Supercapacitor Charger

Becker Sharif, Mimi Petersen, Adam Holman

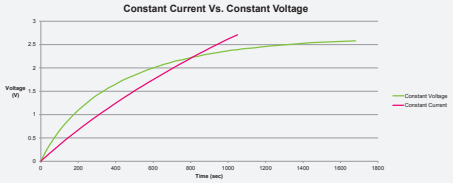
Abstract

Supercapacitors are an energy storage device with an extremely high power density, which gives them an advantage over conventional batteries in their abilities to charge and discharge very quickly. To utilize the function of this emerging technology we created a device with the capability to charge in minutes enough energy to charge an iPhone 60%, something that would take a conventional battery hours. Once quickly charged, this portable device can be unplugged to charge any electronics device that has a USB connectable port.

The final design consisted of a modular system which contains two devices: a 350W switch-mode power supply which outputs 24V at up to 15A and the supercapacitor bank combined with the circuitry which controls the charging of the supercapacitors and the discharging of the supercapacitors to the USB port.

Approach

The approach of our design was split into three subsystems: the charging circuitry, the supercapacitor bank, and the discharging circuitry. We focused initially on how supercapacitors worked and how to efficiently and quickly charge them. We compared the charging times of constant current vs constant voltage.



Graph 1. The comparison of the charging voltages of a 350F supercapacitor charged under constant current and constant voltage

After determining that the 350F supercapacitor provided for the highest energy density at the lowest price point, we connected the supercapacitor bank and turned to the circuits.

We chose a charge controller IC which provided constant current of 20 Amps to minimize charge time to a quick 90 seconds. To provide enough power to transfer so much energy in such little time, we built a 350W power supply, modeled after the Mean Well power supply.

The discharging circuitry was built around a buck/boost converter which could utilize 90% of the available energy in the supercapacitors to charge a mobile device.

After integrating the three subsystems into a final prototype we designed and built PCBs for the circuits and built a final enclosure for the device.

Overview

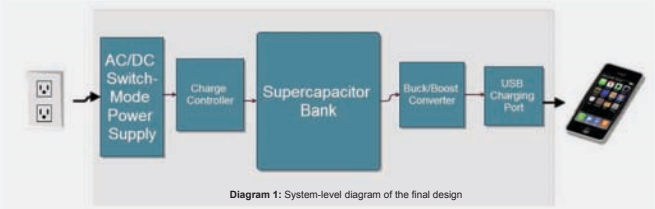


Diagram 1: System-level diagram of the final design

AC-DC Power Supply:

- Input from a traditional 120 VAC wall outlet.
- Switch-mode Half-Bridge topology provides up to 84% efficiency.
- Capable of providing 350 W of power.
- Minimal output voltage ripple provides a stable power input to the charging circuit.

Figure 1.
Power Supply

Charging Circuit:

- Provides for constant current at 20 Amps for decreased charging time.
- Buck converter provides for efficiency up to 94%.
- Control loop regulates output current to an accuracy of 6%.

Supercapacitor Bank:

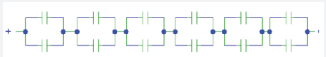


Figure 2.
Supercapacitor Configuration

- Twelve 350F Supercapacitors provide 4.2 Watt-Hours of energy.
- Connected in a parallel/in-series combination to provide maximum voltage of 16.2 V.
- Stores enough energy to charge an iPhone battery up to 60%.

Discharging Circuit:

- Buck/boost converter maximizes efficiency, up to 90%, to use as much energy from supercapacitor bank.
- Converter regulates output voltage to 5V as the supercapacitor bank drops in voltage down to 2.7V.
- Provides 5 V DC @ 700mA to USB output compatible with all mobile devices.

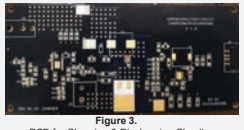


Figure 3.
PCB for Charging & Discharging Circuit

Acknowledgments

We would like to thank Michael Ready for his sponsorship of this project, which included the proposal of the design idea, financial support, and assistance throughout the project.

We would also like to thank David Munday and Ethan Papp for their instruction and guidance.

Design for the power supply was based on a similar design by Mean Well Inc.

Analysis

Our two quantitative goals were to charge the supercapacitor bank in under 3 minutes, and to deliver at minimum 50% charge to an iPhone. In order to achieve these two goals we chose the best configuration of supercapacitors to store energy without reducing the charge time and maximize the efficiency of the discharging system.

Efficiency:

The discharging circuit utilized the LT3115 IC to perform the buck/boost power converter operation, with the following efficiency:

- Efficiency from 16.2 V – 7 V: 90%
- Energy in supercapacitor bank from 16.2 – 7 V: 12451 J
- Efficiency from 7 V – 2.7 V (minimum input for boost): 86%
- Energy in supercapacitor bank from 7 – 2.7 V: 1206 J

Total Energy Stored in the Supercapacitor: 15309J

Energy Delivered: 12451(.90)*1206(.86)=12413J

Total Efficiency of Discharging Circuit: 12413J/15309J= 81%

Results

We created a portable charger prototype that utilizes supercapacitors as the main energy storage device. We charged the supercapacitor bank at 20 Amps in about 90 seconds, using a 350W AC/DC power supply. The supercapacitor bank and discharging circuit provide enough energy to charge an iPhone 4S to 60%.

As supercapacitor technology is improving, we constructed a scalable prototype which can be altered to adapt to changing supercapacitor size and capacitance. Instead of size and cost we focused on high energy storage and a very short charge time, both of which we achieved with our final design.

Conclusion

Supercapacitor technology is continuing to grow and new materials are being used to increase energy density. Materials such as graphene are also being looked at in conjunction with supercapacitor technology to possibly create energy storage devices that are even smaller and lighter than traditional lithium ion batteries with increased storage. Once supercapacitor technology has advanced, our circuitry and design can easily be used to charge the newer, smaller supercapacitor bank. We believe that supercapacitors have not found their place in the market place just yet; however, once they do they will replace batteries as the superior energy storage alternative.

soe.ucsc.edu

Email us:
fhowley@ucsc.edu

Join us on Facebook:
facebook.com/BaskinSchoolofEngineering

Patrick Mantey
Associate Dean, Industry Programs
CITRIS Campus Director
Director of ITI
Jack Baskin Endowed Professor, Computer Engineering
mantey@soe.ucsc.edu

Tim Bensch
Director of Corporate Development
tbensch@ucsc.edu

Frank Howley
Senior Director of Corporate Development
fhowley@ucsc.edu

To view all project posters, please go to this link:
http://csspp.soe.ucsc.edu/posters/2014

UNIVERSITY OF CALIFORNIA
SANTA CRUZ

Baskin Engineering
UC SANTA CRUZ