Project C.A.S.L.A: Calculator for American Sign Language Arithmetic

Greta Perez-Haiek, Aadhav Sivakumar, Nathan Smurthwaite

December 16th, 2024

1 Abstract

Project C.A.S.L.A (Calculator for American Sign Language Arithmetic) seeks to create a glove for ASL users to seamlessly translate physically signed arithmetic into readable text using Parallax's Basic Stamp 2 (BS2) and other relevant electronic components. The design consists of 5 flex sensors attached to each digit of a glove to detect the bend of each finger. The resistance generated from the flex sensors when the gloved hands sign numbers acts as an analog input, which can be read for further ASL interpretation. Four buttons, one for *addition*, *subtraction*, *next digit*, and *equality*, acts to capture the current desired arithmetic operation. An accompanying LCD display keeps track of the arithmetic operation (and displays results) for user monitoring. When calibrated to a user's hand, C.A.S.L.A showcases the ability to detect all 10 ASL numbers and sequentially conduct calculations (akin to a hand-held calculator). This project will benefit the ASL community as it allows ASL users to interact with those who are not fluent in ASL in a mathematical context, and vice versa. When conducting conversations that utilize mathematical components and arguments, not every individual might have access to a pen and paper at hand, and not every American-English Speaker understands the numerical ASL hand signs. Utilizing C.A.S.L.A can enable more fluent conversation in mathematics between the hearing and deaf communities.

2 Introduction

Within the United States, approximately 15 percent of the adult population has reported some or total hearing loss, with over 11 million people experiencing total deafness or extreme hearing difficulties [4][6]. On the other hand, according to the "Commission on the Deaf and Hard of Hearing" of the State of Rhode Island, more than a half-million people throughout the United States use ASL to communicate as their native language [1]. While deafness as a condition has prevailed throughout human history, ASL (as a method of communicating) is a relatively new language, first appearing in the 19th century with the founding of the first successful American School for the Deaf. Ever since, ASL is slowly growing in popularity, especially with the recent inclusion of ASL as a secondary language option within secondary education.

While the inclusion of ASL in modern-day schooling has enabled non-deaf individuals to learn and connect with the deaf community, the previous statistics have demonstrated a communication gap between the deaf and the hearing. Language rejuvenation programs in and out of schooling have empowered many to take the first step in learning ASL, but there still lacks a structured approach within ASL for communicating complex mathematical concepts.

Since ASL was developed as a day-to-day communication platform, expressing abstract ideas such as mathematical concepts were a difficulty. According to a 2016 cross-examination "On the Expression of Higher Mathematics in American Sign Language", researcher John Tabak noted that "even those fluent in American Sign Language (ASL) and with a strong background in mathematics can find it difficult to express mathematical language in ASL, either because so many common mathematical words have no counterparts in ASL or because the grammar of mathematics is so different from the grammar of ASL" [5]. The difficulty to participate in advanced mathematical conversation has extended beyond deaf mathematicians, as deaf school children have been noted to struggle with simple mathematical concepts, tending to showcase a "a gap of approximately three years behind their hearing peers in mathematic performances" [3]. Needless to say, there persists a problem within the ASL community to express themselves mathematically and for the hearing to express mathematical concepts back.

The goal of project C.A.S.L.A is to enable users to communicate and conduct arithmetic using sign language. This way, ASL speakers can learn to associate arithmetic operations with numerical hand signs, and communicate arithmetic in conversation with a hearing individual without the need to pause the conversation and write down concepts. This glove will also have the ability to conduct calculations in real time, enabling streamlined, convenient, and accurate mathematical conversation. An extended utility of the project can involve acting as an alternative, "hands-off" calculator for those that might not be able to use a hand-held calculator, such as the disabled or the elderly. The purpose of this project is to ultimately help individuals to learn, use, and benefit from mathematical concepts without compromising method of communication, regardless of whether a user is deaf or not.

3 Literature Review

3.1 The History of the ASL Glove

So far, not much has been done to address the gap between ASL and math, as recognizing signs for abstract topics like math is a relatively new phenomenon. However, plentiful of ASL-to-English gloves have been invented to combat the language communication gap between the hearing and the deaf. The first one was notably invented in 1981 by a Bell Labs engineer named Gary Grimes: This glove, originally designed for data entry using the 26 manual gestures of the American Manual Alphabet, later inspired Stanford University researchers James Kramer and Larry Leifer in 1988 to invent a "talking glove" that could make interactions between deaf and non-deaf people easier.

As the decades rolled on, the idea of bridging the communication gap between deaf and hearing individuals extended past the spoken word and towards the digital age. Ryan Patterson, an eighteen-year-old fond of communicating with his friends through instant messaging, designed a sign language translator glove in 2002 that translates ASL to English, then wirelessly transmits the data to a portable device that displays the text on-screen. 14 years later, Navid Azodi and Thomas Pryor, two students from the University of Washington, invented the "SignAloud" gloves that could translate ASL into speech or text, merging developments from Ryan Patterson, James Kramer, and Larry Leifer. Consequentially, their invention won the $10,000 Lemelson-MIT Student Prize, showcasing the scientific community's recognition and support for innovations that promote accessibility.

In 2019, A young engineer from Kenya, Roy Allela, created "Sign-IO", gloves that allow the deaf to communicate with those who do not know sign language. Android App Enabled text-to-speech capabilities differentiated "Sign-IO" from predecessors: Now, anyone with a smart-phone can benefit from ASL-to-English technology! As a result, "Sign-IO" was the grand winner of the Hardware Trailblazer award at the American Society of Mechanical Engineers (ASME) global finals in New York, and placed 2nd runner-up at the Royal Academy of Engineering Leaders in Innovation Fellowship in London. The overwhelming success of these ASL gloves highlighted the importance of day-to-day app integration in a commercialize-able product. This way, individuals would be able to adopt these gloves into their daily lives without hauling bulky computers around.

3.2 Merging ASL with AI

With the emergence of artificial intelligence comes the desire to integrate smart algorithms into everyday products, and the ASL gloves offer no exception. In 2020, UCLA researchers decided to improve upon the standing success of ASL gloves, and thus, experimented with merging applied machine learning algorithms to extend the vocabulary base of the glove [7]. These "smart-gloves" have the capability of learning new words based upon gesture recognition patterns alone, with a "demonstrated recognition rate of up to 98.63 percent with an elapsed time of less than a second." Improving upon the recognition accuracy and timing of the ASL gloves has been met with great success amongst the ASL and scientific community, resulting in a publication. While the extent of the potential behind integrating machine learning with the ASL glove approach is still being examined, the future holds much potential in integrating inclusivity with innovation for the ASL community.

3.3 The Problem with Contemporary ASL Gloves

The previous iterations of the ASL glove have been revolutionary in improving communication methods between the deaf and hearing communities. However, there has not been much progress in integrating mathematical lingo into these gloves. The smart-gloves gloves invented by UCLA (notably the cutting edge of ASL-to-English technology), does not have the ability to do arithmetic or incorporate mathematical concepts in their lexicon. The complexity of language has preoccupied researchers in their plight to bridge the communication gap, neglecting the field of mathematics as a whole as an avenue for improvement. As far as "touchless" calculators go, some might argue that voice-to-text large language models could potentially perform mathematical calculations and bridge the communication gap. The ethics surrounding the inclusion of large language models in day-to-day conversation, however, proves this approach to be undesirable. Large language models tend to be trained upon web-scrapped data, which may not be mathematically correct or ethnically sourced. Including a large language model in conversation might also be frustrating when it misinterprets a formula or the intentions of a word. The solution to incorporating mathematics in daily ASL-to-English and English-to-ASL conversation must be of an analog variety: simple, cheap to assemble, accessible, and most importantly, enables the user to have full-control of their expression!

4 The Solution: C.A.S.L.A

4.1 Introducing the Solution

C.A.S.L.A is a glove that can effectively register mathematical expressions, in the form of inputs from an ASL user, and displays the result with an LCD. The functionality of the glove allows the user to input larger multi-digit numbers and do basic operations with them. The operations that the glove support is 4 digit addition, subtraction, multiplication, and division expressions, and can produce results accurate to the thousandths place. This "hands-off" calculator approach allows ASL individuals to translate their mathematical ideas for non-ASL users to comprehend, not to mention aid those in producing moderately difficult arithmetic results without pausing the conversation to pull out a calculator or a piece of paper and pencil. This solution has not been implemented before, as the communication of mathematics in ASL have not been a focus of the engineering community. C.A.S.L.A also came out to be a far more financially responsible option for ASL users, as current market products for ASL-to-English conversion (such as the BrightSign Glove), are thousands of dollars. Hopefully, this approach will help others learn and understand each other's mathematical ideas without compromising language fluency.

4.1.1 How to use C.A.S.L.A

To use C.A.S.L.A., the user first puts on the glove, ensuring the wires run along the top side of the wrist. Once the fingertips are properly positioned, the number being signed will appear on the LCD screen. To add another digit to the number, press button 4: this will lock in the previous digit. After completing the first number, select an operator.

Quickly pressing the first button initiates the addition process, while quickly pressing the second button starts subtraction. Holding the first button for more than a second will start multiplication, and holding the second button will initiate division.

Next, the second number can be signed following the same procedure as the first number. Both numbers are limited to 4 digits, as larger numbers exceed the math capabilities of the Basic Stamp 2.

Once both numbers are entered, pressing button 3 completes the calculation. The result will display on the second line of the LCD for a few seconds before the system reverts to its original state, ready for a new first number input.

If an emergency happens, such as spilling water on oneself, there is a red kill-switch contained in the C.A.S.L.A box that would instantly terminate the program and cut power. One should not utilize the glove when eating or drinking, or press multiple buttons at the same time, as it would damage the circuitry.

4.2 Constructing the Glove

The glove itself is a standard wool winter glove, where each finger is fully encased in fabric. Each of the fingers has a flex sensor (designed to produce circuit resistance when bent) rigidly affixed to the top of each finger with a tape. The security in design around the flex sensors ensure that the sensors don't go out of place while signing, or get damaged in use. The flex sensor aligns itself on top of the fingers, and has external wires that lead to an Resistor-Capacitor circuit for each finger. There are additional rubber bands to pull the flex sensor back when the fingers curl and uncurl, allowing for more consistent readings from the circuits. The flex-sensor readings, which would be read by the BS2 using the programming command RCTime, will be incorporated into an algorithm that would recognize what number the glove-wearer is signing. There are various buttons that refer to the type of arithmetic that one would want to conduct, such as addition, subtraction, multiplication, and division. An agitator, attached to the palm of the glove, spins in proportion to the size of the signed numbers and results. This feature is incorporated after further consideration for those that might also have difficulty seeing and would prefer a tactical communication method as well.

The entire glove's circuitry is encapsulated in a transportable box, although later improvements would include compacting the circuits so that one can transport the entire system on the surface of one's wrists (See Discussion and Conclusion).

4.2.1 Bill-Of-Materials

Part name	link	price	quantity	totals
flex sensor 100mm	https://www.ama	10	2	20
flex sensor 100mm (2)	https://www.ama	10	3	30
Glove		2	1	2
Various capacitors and resistors		5	1	5
LCD/Speaker	https://www.para	35	1	35
Basic Stamp 2	https://www.para	50	1	50
Board of Education	https://www.para	70	1	70
				sum:
				212

Figure 1: Bill Of Materials

Referring to Figure 1, the total prototype cost is \$212. This seems expensive, but the leading commercial alternative is Brightsign, which costs \$3100 (See Appendix 8.2). BrightSign, like many other smart gloves, has the ability to translate a variety of sign language libraries, including ASL. Despite being Market-ready (and having the potential to be taught mathematical abstraction if a mathematical lexicon library were to exist), the steep price-tag of \$3100 price is a large barrier of entry for everyday users. No mathematical libraries currently exist in their database, making it an incomplete product for those seeking to streamline their mathematical conversations. Not to mention the incomplete skill-range, the price point is far outside of the purchasing range for a middle-class consumer.

Since the C.A.S.LA prototype is significantly cheaper than BrightSign, customers would be more likely to purchase it due to convenience, focus in scope, and accessibility. The parts are cheap to purchase and easy to replace, and the mechanics of the glove are simple to learn. The PBasic Language is also Open Sourced and easy to diagnose if a problem were to persist. Nevertheless, if C.A.S.L.A were mass-produced, alternative design choices would be made (for the sake of cost-efficiency).

4.2.2 Cost Analysis for Potential Mass Production

Despite the ease of the utility behind the BS2 and associated PBasic language, there are alternative design choices that can be made to reduce the cost per individual item. In mass production, the flex sensors and agitators (another significant cost) could be ordered in bulk, leading to significantly decreased costs. However, the biggest financial burden associated with the design happens to be the BS2 and the Board of Education (BOE), a circuit board where the bulk of the circuit is built upon. The BOE extends the utility of the IO pins and adds and easier way for the user to connect power, both of which can be recreated using alternative approaches. The Stamp itself can also be replaced with a cheaper alternative, such as the Raspberry Pi Zero, which has more functionality at a fraction of the cost ($15). Since the Board of Education and the BS2 encapsulates more than half of the production bulk of the prototype (see Appendix 8.1), replacing those can allow an increased profit if C.A.S.L.A were to become a commercialized product.

Commercializing C.A.S.L.A would succeed after conducting a thorough customer discovery within the ASL speaking community, and their willingness to invest in a product that would make communicating mathematics easier. Only after researching the experiences of the ASL community in communicating, learning, and expressing mathematics can a company estimate whether C.A.S.L.A will be a successful product.

4.3 Circuitry

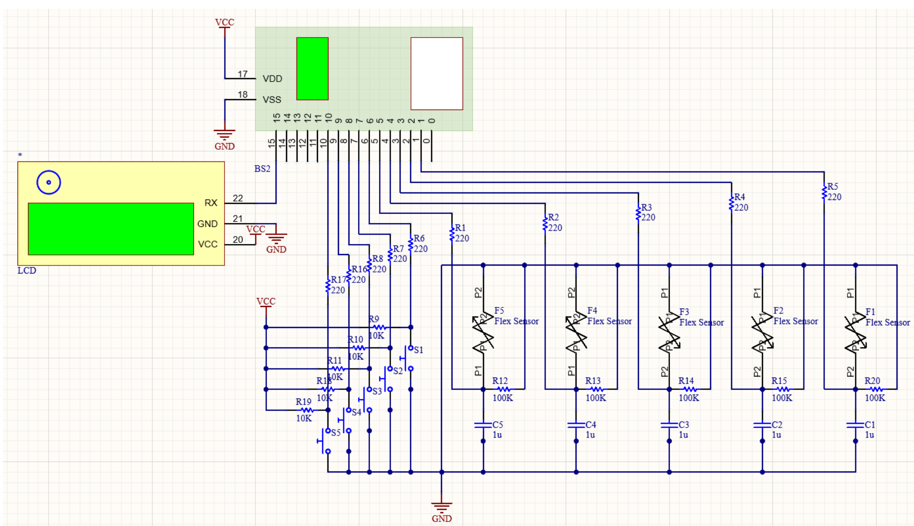


Figure 2: schematic of connections to and from the BS2

How does the glove work? The bulk of the science behind the glove is associated with the RC circuits linked to each finger. The RC circuits (see Figure 2) are linked to each finger with the flex sensor acting as the variable resistor value being measured. When the flex sensor is fully extended, it generates a very high resistance: if the resistance is too high in an RC circuit, the capacitor cannot properly discharge itself in a reasonable amount of time. This phenomenon leads to an RCTime reading overflow, as the limit capabilities of the BS2 when reading a discharge is approximately 131 ms, and straightening one's fingers would push the capacitor past that limit. To solve this, a 100k Ohm resistor was placed in parallel with the RCTime circuit, so that even when the flex sensor's resistance is near infinity, it still allows the capacitor to discharge near (but not over) the max time of 121 ms. Alternatively, when the fingers are fully flexed, the flex sensor's resistance gets orders of magnitude smaller than the 100k Ohm resistor, hence becoming negligible in the RCTime calculations.

Contained in the circuit, there are also 5 switches: 2 for the arithmetic operator selection, 1 for adding an extra digit to a desired hand-signed number, 1 to get the results of the mathematical expression, and 1 as the kill switch. A kill-switch is the utmost importance when designing wearable circuitry, as having an alternative method of halting the program's functionality is crucial for sustaining safety in emergencies, such as spilling water on oneself.

All of the switches are in pull up/active low configuration to ensure that there are no floating voltages on the input pins, hence minimizing the button-bounce phenomenon. There is also one pin dedicated to the LCD, which can be used with SEROUT to fully control text-output, and another pin dedicated to the "agitator" (in the form of a motor), which uses the PULSOUT command to control it's speeds.

4.4 Programming

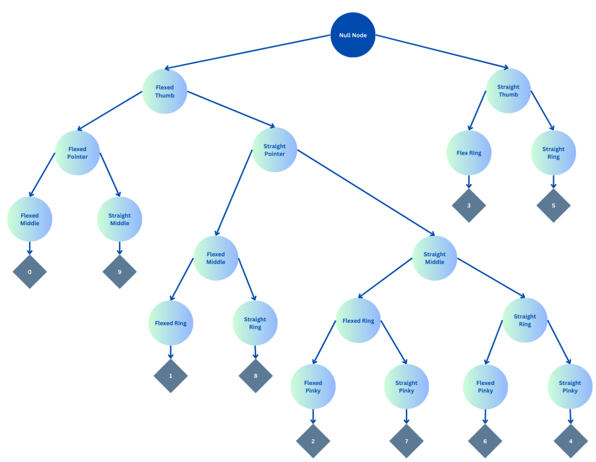


Figure 3: Decision tree for selecting number

With accessibility being the goal, C.A.S.L.A must account for the unique and varying dynamics of the human hand. Once the glove is slipped onto the user, the next step was to interpret and quantify the user's unique finger movement through a sensor through a process called ”tuning.”

The first half of the programming focuses on processing inputs from the glove. The compiled PBasic code requires storing each finger's ”live state” as five words of RAM, with each word storing the position of a specific finger. Thorough testing confirmed that each finger's ”live state” response is not linear, hence, the program is designed to treat each finger as a binary value of either ”curled” or ”uncurled.” This programming design choice eliminated the need for linearization.

To tune the thresholds between these two states to a user's unique hand-size, each finger should be slowly curled until it is comfortable in the ”completely curled” position for a user. A user should not force this position- it should be a natural state, and the C.A.S.L.A glove would attune to it. The RCTIME output at that point should be recorded as the threshold for each finger, and then later on set as the value that differentiates a finger being ”up” from being ”down.” Although this process was time-consuming, it resulted in better and more accurate responses, particularly for users with varying hand sizes. A *state machine* was implemented to control the program's flow, as shown in Figure 4.

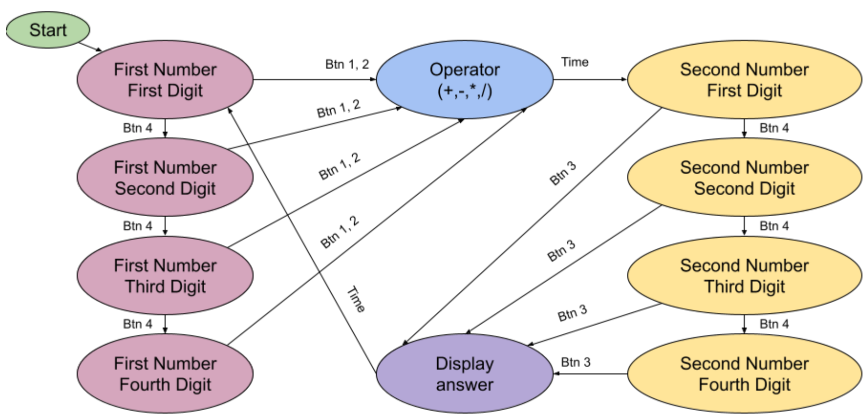


Figure 4: State machine for code control

The *state machine* dictates the flow of the program, with all possible transitions between states being directly controlled by the buttons. The multiplication operator can be selected by holding the "+" button down for more than a second, while the division operator can be selected by holding the "-" button down for more than a second. After entering the first and second numbers, pressing the "=" button displays the final answer. The program then returns to its original state, ready to display than evaluate for the next mathematical expression!

5 Results and Demonstrations

After a thorough research and development period, substantial results were obtained from the developed prototype. While the BS2's hardware and programming capabilities are limited, with a creative design and code implementation, the memory can be maximized to its absolute potential. In this instance, the BS2 is able to receive 5 analog sensor signals, 5 digital sensor signals, output to one actuator, and provide an LCD display with 3 General Purpose Input Output (GPIO) pins still available. Given 5 unique analog signals and with some provided direction from digital inputs, the BS2 is now capable of translating a unique grammar structure from a language that is not typically capable of supporting mathematical arithmetic.

The results of this prototype show that, with sequential captures of 5 separate analog signals, it is capable to perform mathematical operations such as addition, subtraction, multiplication, and division with 2 numbers that can each be up to the thousands place. The subtraction operator handles negative numbers appropriately and the division operator supports decimals up to the thousandths place. Multiplication is handled up until the computational limits of the BS2.

Feedback is also given back to the user based on the analog signals received as a result of the user's movements. This feedback is in the form of a servo actuator within the palm of the glove and an LCD Display. To control the motor, you send multiple pulses that last for 20 ms each, with the duty cycle of the pulse determining the speed of the servo. A high time of 1.5 ms corresponds to the continuous servo being stopped, while a high time of 1.7 ms corresponds to the servo going full speed ahead. For the glove, the servo speed is a result of the magnitude of the number that is translated from the analog signal combination. This provides the user with some feedback as to if the number signed is being correctly displayed without having to manually view the LCD. The LCD is another form of feedback that is geared toward the recipient as it correctly and iteratively displays the process of the mathematical operation that is being communicated by the user as well as the output.

6 Discussion and Conclusion

While what was achieved here with the BS2 does produce a working prototype, if research into this topic were to continue then certain improvements would need to be in place, Firstly, there is the choice of the BS2 as the micro-controller. The BS2 has its advantages in terms of error diagnosability and dexterity. However, to scale up the capabilities, speed, cost, manufacturability, reserved memory, and flexibility of this prototype in future iterations, a different micro-controller should be used as the basis. Recommendations include Arduino Nano, Pi Zero, and Raspberry Pi Pico. With a different micro-controller, more improvements can be made.

For instance, with more GPIO pins available, in place of the RCTIME command and accompanying circuits for analog signal detection, the Integrated Circuit (IC) ADC0831 would be in use to detect each analog signal. The ADC0831 would allow the response of the flex sensor to be linearized. In its current state with RCTIME, the response is exponential. On the mechanical side of the prototype, in place of elastic bands to maintain tension on the flex sensor, metal springs would be used. It was realized during testing that given the frequency of movements of the human hand during ASL communication, the fatigue life of each band was reached at an accelerated rate. Therefore, switching to metal springs would vastly improve the longevity of the prototype as the springs are less likely to loose tension. Continuing with mechanical improvements, the encasing for the board and controller could be minimized in future installments. Ideally, the breadboard used would instead be a printed circuit board (PCB) and given the size of the micro-controller the entire circuitry component could be minimized and consolidated to fit on the back of the palm of the user during operation. The PCB would be relatively cheap because only a single side of copper would be necessary considering the simplicity of both the RC time circuits and the button circuits with the pull-up resistors attached.

Finally, more complex mathematical concepts, as well as the inclusion of the ASL Alphabet and small phrases could be implemented to allow a more flexible conversation between user and recipient.

7 References

[1] “American Sign Language.” American Sign Language — Commission on the Deaf and Hard of Hearing (CDHH), cdhh.ri.gov/information-referral/american-sign-language.php. Accessed 15 Dec. 2024.

[2] “How Many Deaf People Live in the United States?” National Deaf Center, 28 Mar. 2024, nationaldeafcenter.org/faq/how-many-deaf-people-live-in-the-united-states/.

[3] “Mathematics Differentiated Instructional Strategies - Deaf and Hard of Hearing.” TTAC Online, ttaconline.org/differentiated-instructional-strategies-deaf. Accessed 15 Dec. 2024.

[4] “Quick Statistics about Hearing, Balance, and Dizziness.” National Institute of Deafness and Other Communication Disorders, U.S. Department of Health and Human Services, www.nidcd.nih.gov/health/statistics/quick-statistics-hearing. Accessed 15 Dec. 2024.

[5] Tabak, John ”On the Expression of Higher Mathematics in American Sign Language,” Journal of Interpretation, 2016: Vol. 25: Iss. 1, Article 10

[6] Young, Michelle et al. “How Many People Use Sign Language? A National Health Survey-Based Estimate.” Journal of Deaf Studies and Deaf Education, U.S. National Library of Medicine, pubmed.ncbi.nlm.nih.gov/36423340/. Accessed 15 Dec. 2024.

[7] Zhou, Z., Chen, K., Li, X. et al. Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays. Nat Electron 3, 2020, 571–578. https://doi.org/10.1038/s41928-020-0428-6

8 Appendix

8.1 Project Presentation

C.A.S.L.A

Calculator for American Sign Language Arithmetic

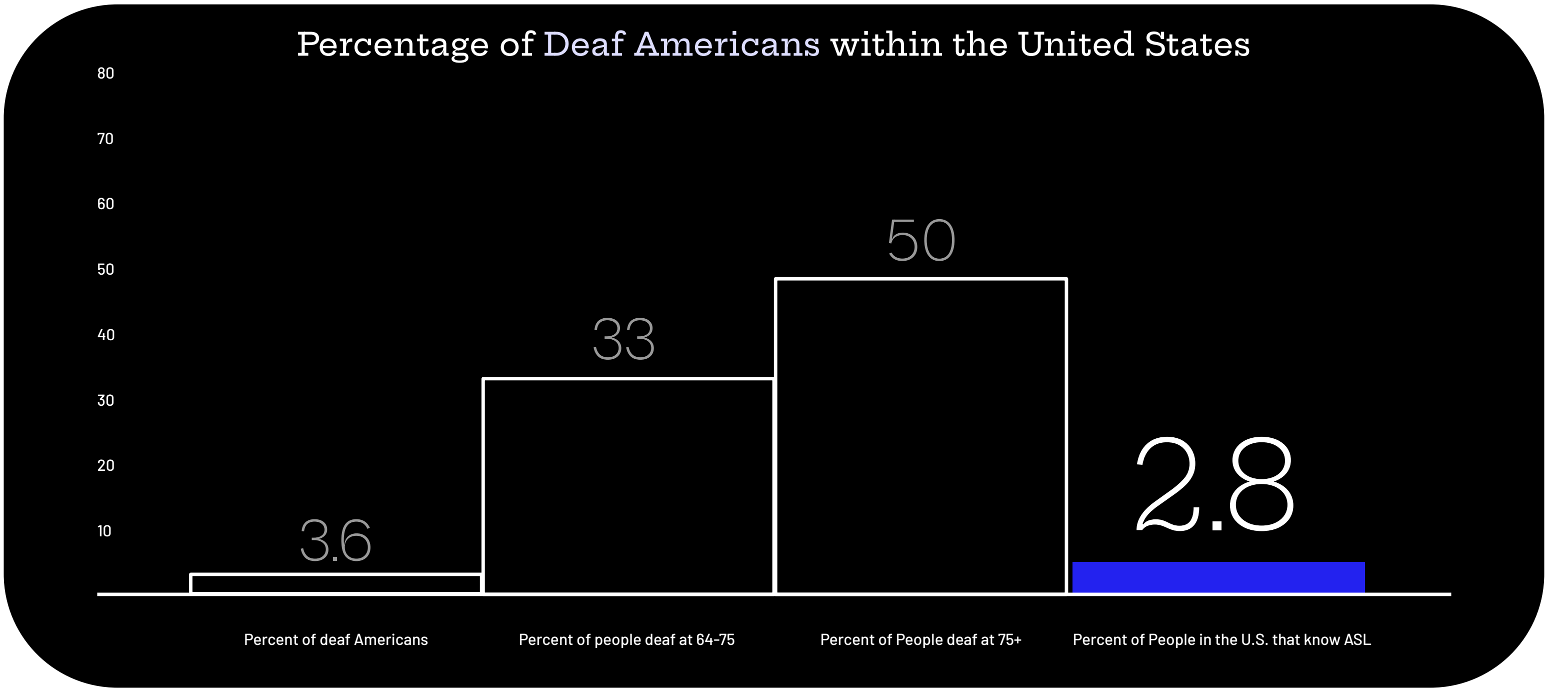
Mechatronics F24
Prof. Kapila

Greta Perez-Haiek, Aadhav Sivakumar, Nathan Smurthwaite

01

What's the Problem?

There is a communication gap between the deaf and the hearing...



Source: https://www.oliveunion.com/ustest/hearing-loss/hearing-loss-statistics-in-the-united-states/

... especially when it comes to **mathematical** concepts.

> "*Even those fluent in American Sign Language (ASL) and with a strong background in mathematics can find it difficult to express mathematical language in ASL, either because so many common mathematical words have no counterparts in ASL or because the grammar of mathematics is so different from the grammar of ASL…*"

- John Tabak (2016)

On the Expression of Higher Mathematics in American Sign Language

Previous Solutions address the language communication gap...

Sign-to-Speech Translation using Machine-Learning-Assisted Stretchable Sensor Arrays

University of Los Angeles (UCLA)
Zhou, Z., Chen, K., Li, X. et al
2020

Issues

Enhances vocabulary and accuracy using a Multiclass SVM Algorithm. Despite having massive potential, it is still in an experimental phase. While it has the ability to learn numbers, it does not include mathematical capabilities such as calculations or expressing arithmetic.

The BrightSign Glove

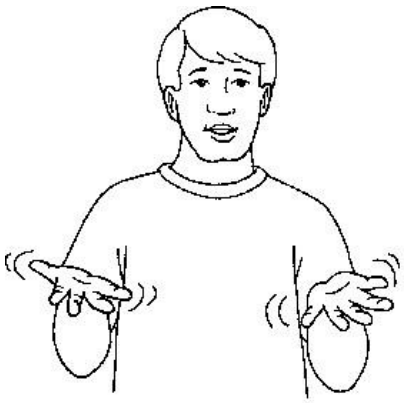
"The only sign language glove in the world that can translate any sign into any spoken language - instantly."

https://www.brightsignglove.com/

Issues

Has the ability to translate a variety of sign language libraries, including ASL. Despite being Market-ready (and having the potential to be taught mathematical abstraction), the steep price-tag of $3100 price is a large barrier of entry for everyday users. No mathematical libraries currently exist in their database.

But nothing is done about the lack of mathematics in ASL.



"What?!" in ASL

02

What’s the Solution?

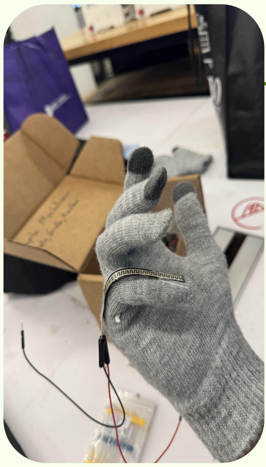
Introducing C.A.S.L.A:

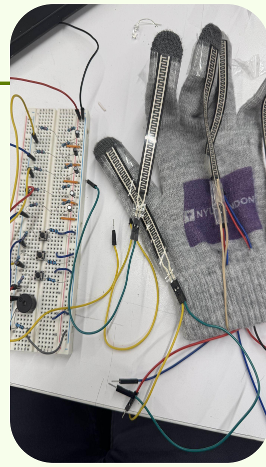
Calculator for American Sign Language Arithmetic

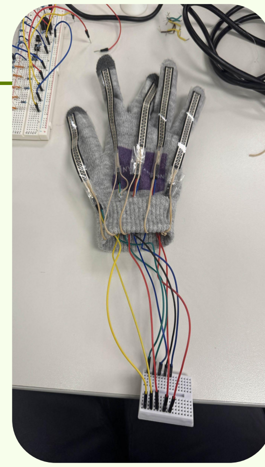
- Effectively register inputs from ASL users.
- User has the ability to input multi-digit numbers and do basic operations.
  - Addition, Subtraction, Multiplication, and Division
- Cannot handle multiplication of numbers that exceed the range of the number system provided on the BS2 (the resultant size of a Word).
- Can handle fractional division up to 3 decimal places (thousandths)
- Gives vibrational feedback based on the current number being signed
  - The higher the resultant number, the faster the agitator spins.
  - This helps blind/deaf users understand numerical values through tactical simulation
- Can be calibrated to one’s personal hand for maximum accuracy.

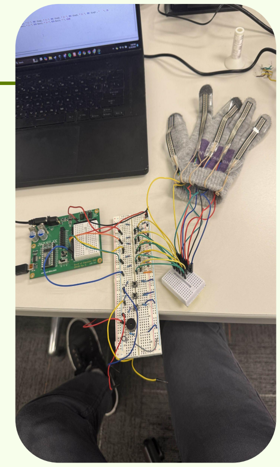
Glove Construction

This slide outlines the process used to construct the glove step-by-step









Step 1

Testing the curvature of a singular flex sensor, and the corresponding rctime values

Step 2

Attaching more flex sensors and adding tensioning rubber bands to reset flex sensor positions

Step 3

Connecting all flex sensors to individual rctime circuits on a breadboard

Step 4

Organizing the wires coming out of the hand

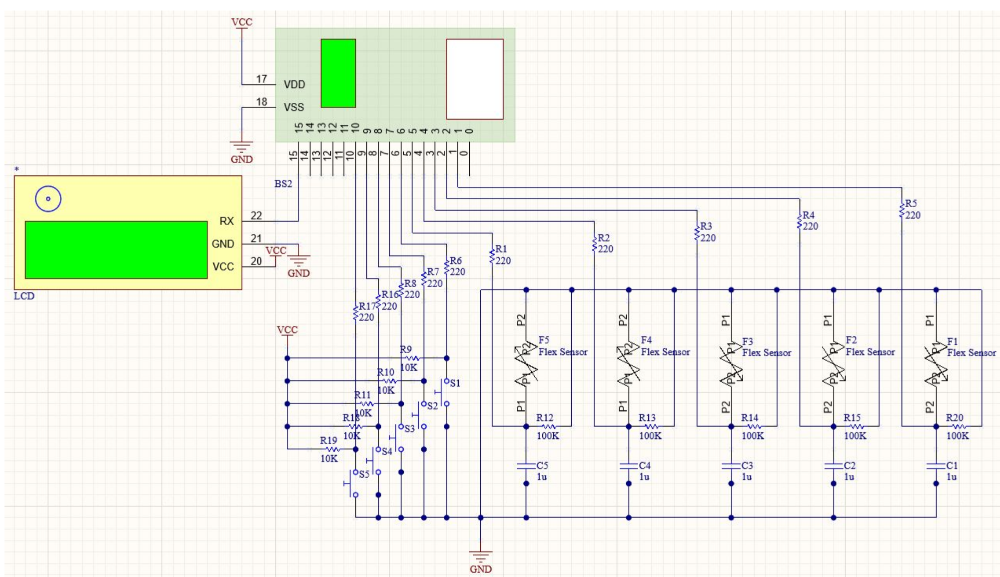
Step 5

Connecting the breadboard with the rctime circuits and the buttons to the basic stamp 2

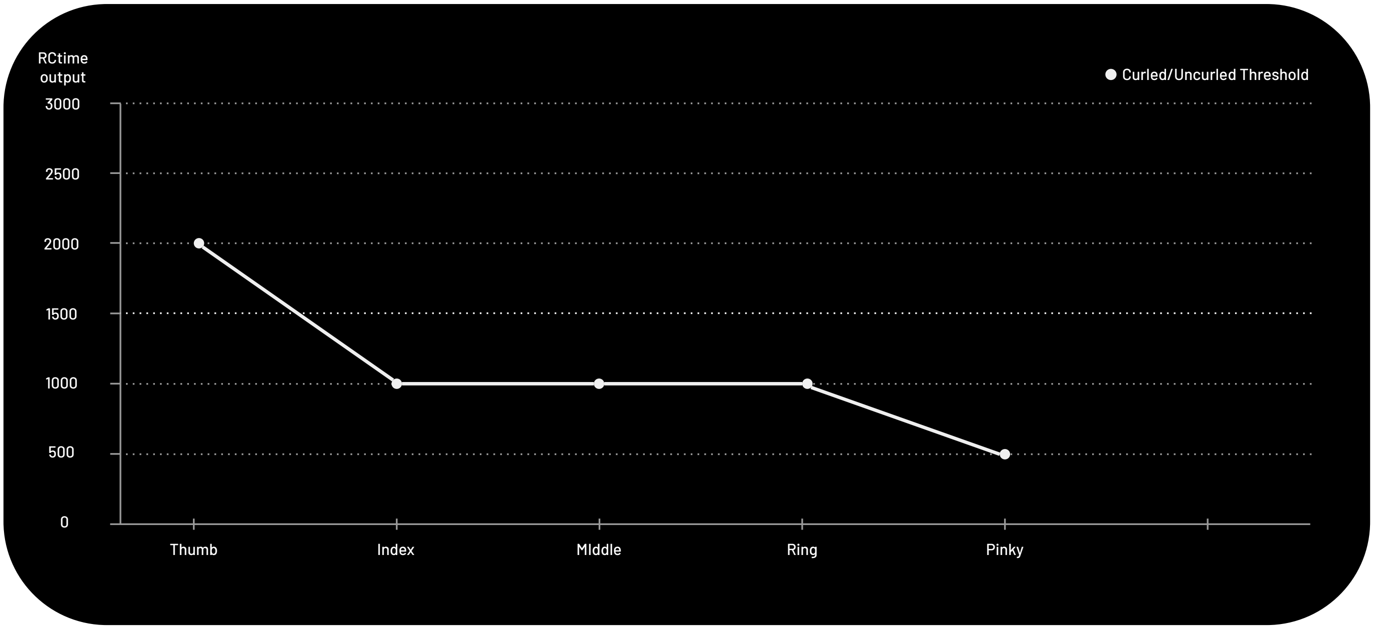
Circuit Diagram

Here is the circuit diagram used to control the project. The sections include:

- An array of 5 flex sensors
  - Connected to a capacitor to function in an RC TIME circuit with the bs2
  - When unflexed, the sensor has Megohm resistance, so a 100k resistor is placed in parallel to ensure that rctime doesn't overflow and cause additional problems in calculation due to size of a word in BS2
- 5 buttons
  - Two of the buttons are used for operator (+,-,*,/)
- LCD and buzzer
  - This combined module is fully controlled with a single pin using the SEROUT command

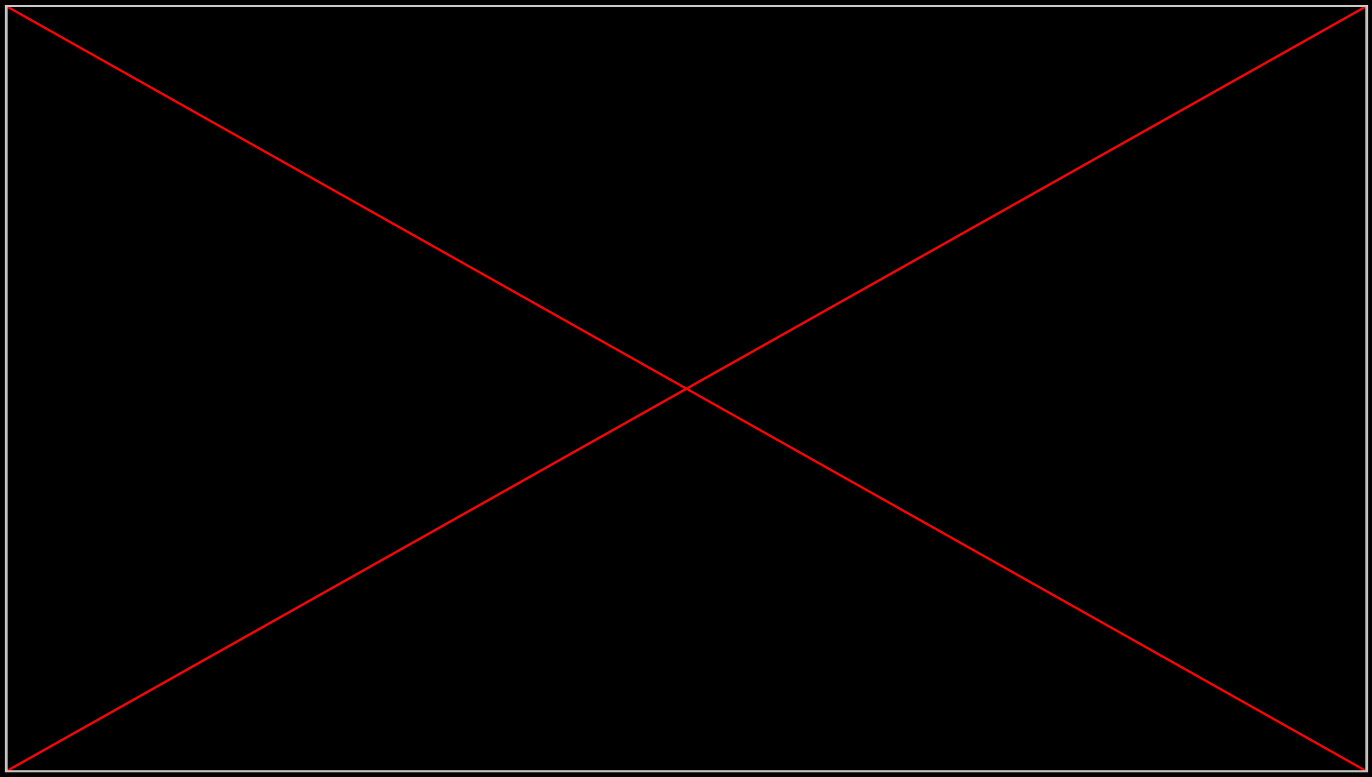


The Tuning Process



When the finger slowly curls inward, the resistance in the flex sensor slowly decreases and rctime reports a smaller value. When low enough, it can be registered as a curled finger, but each finger has a different threshold that can be considered "curled" vs. "uncurled"

The Magic of Tuning: A Video Demonstration of ASL Number to Text Recognition



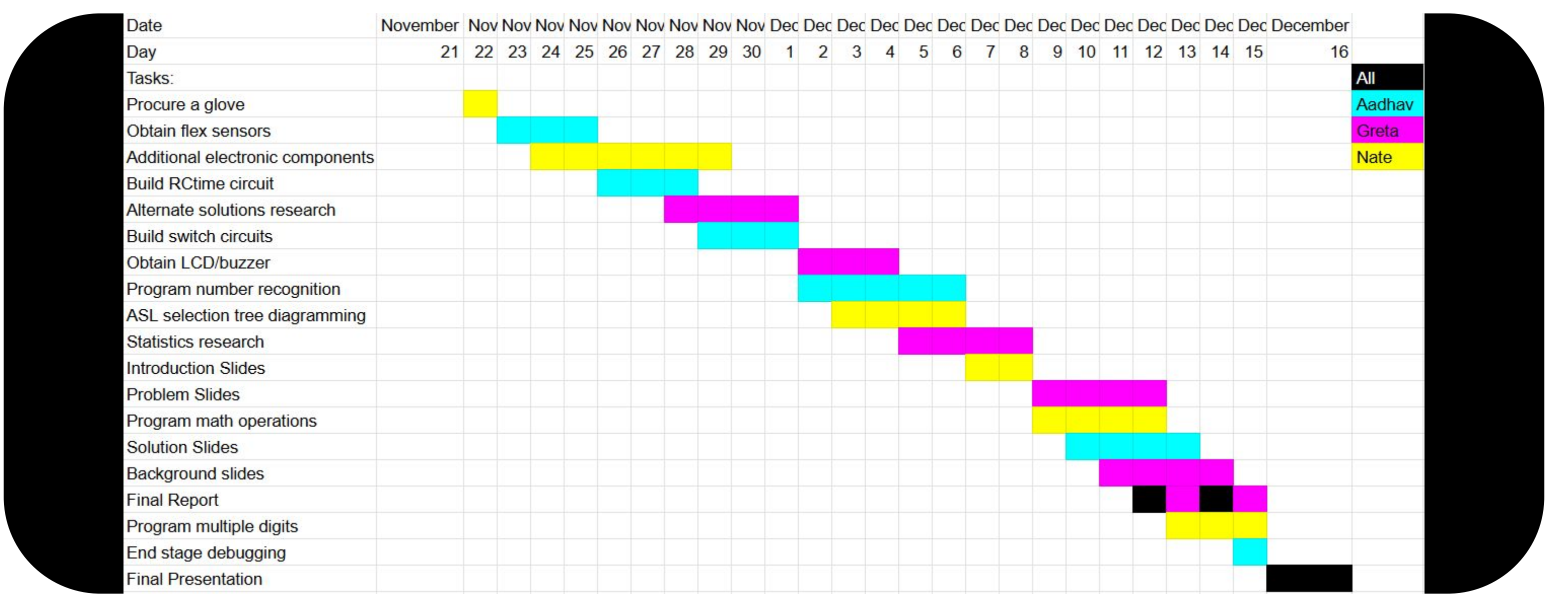
Cost Projection

Here is an itemized list of parts we used to create this project:

- Glove ($2)
- Flex sensors ($10 each, $50 total)
- Various capacitors and resistors ($5)
- LCD/Speaker ($35)
- Basic Stamp 2 ($50 for BS2, $70 for Board of education, $120 total)

This comes out to a total of **$212**, but the majority of the cost comes from the hardware used as part of the class. If tasked to put this into mass production, we would select a chip with basic I/O functionality without the extra board of education I/O extensions, soldering it directly to a PCB. This can be achieved with a PI zero, which has more pins and memory/RAM than the basic stamp 2 at about 1/10th of the price ($120 vs $15).

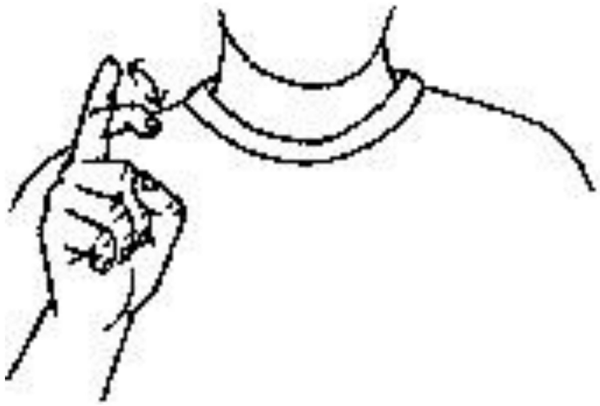
Developing the MVP: GANTT Chart



Areas of Improvement

- If we have more pins, we would use ADCs for glove-to-BS2 communication, instead of RCTime.
  - This way, we can get the exact curvature of the finger instead of binary "on/off" for each finger.
- The flex sensors could have been secured more, but doing that also reduces its ability to straighten back out.
  - Taking the time to engineer channels for each flex sensor, or consider implementing alternative tactical sensors, might be a solution.
- Incorporate complex mathematical concepts, and perhaps letter-variables too.
- Flesh out the transportability of the design- have all circuitry contained on the wrist.

Thank you for listening. Questions?



"Question-Mark" in ASL

References

[1] "American Sign Language." American Sign Language | Commission on the Deaf and Hard of Hearing (CDHH), cdhh.ri.gov/information-referral/american-sign-language.php. Accessed 15 Dec. 2024.

[2] "How Many Deaf People Live in the United States?" National Deaf Center, 28 Mar. 2024, nationaldeafcenter.org/faq/how-many-deaf-people-live-in-the-united-states/.

[3] "Mathematics Differentiated Instructional Strategies - Deaf and Hard of Hearing." TTAC Online, ttaconline.org/differentiated-instructional-strategies-deaf. Accessed 15 Dec. 2024.

[4] "Quick Statistics about Hearing, Balance, & Dizziness." National Institute of Deafness and Other Communication Disorders, U.S. Department of Health and Human Services, www.nidcd.nih.gov/health/statistics/quick-statistics-hearing. Accessed 15 Dec. 2024.

[5] Tabak, John "On the Expression of Higher Mathematics in American Sign Language," Journal of Interpretation, 2016: Vol. 25: Iss. 1, Article 10

[6] Young, Michelle et al. "How Many People Use Sign Language? A National Health Survey-Based Estimate." Journal of Deaf Studies and Deaf Education, U.S. National Library of Medicine, pubmed.ncbi.nlm.nih.gov/36423340/. Accessed 15 Dec. 2024.

[7] Zhou, Z., Chen, K., Li, X. et al. Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays. Nat Electron 3, 2020, 571–578. https://doi.org/10.1038/s41928-020-0428-6

8.2 Project Code

```
' {$STAMP BS2}
' {$PBASIC 2.5}
fing1 VAR Word
fing2 VAR Word
fing3 VAR Word
fing4 VAR Word
fing5 VAR Word

Butt1 VAR Bit 'add
Butt2 VAR Bit 'subtract
Butt3 VAR Bit 'equal
Butt4 VAR Bit 'next digit
SafeButt VAR Bit 'Kill Switch

binaryfinger1 VAR Bit 'pinky
binaryfinger2 VAR Bit 'ring
binaryfinger3 VAR Bit 'middle
binaryfinger4 VAR Bit 'index
binaryfinger5 VAR Bit 'thumb

temp VAR Word
n CON 10
counter VAR Nib
op VAR Nib
aslnumber VAR Nib
b VAR Nib

finalmathanswer VAR Word
finalr VAR Byte
digi2store VAR Word
digi1store VAR Word
diginstore VAR Word
controlstate VAR Nib
idx VAR Nib
controlstate = 1
op = 1
digi1store = 0
diginstore = 0
digi2store = 0
counter = 3
b = 0
'1 is waiting for first number
'2 is adding (and waiting for second number)
'3 is subtracting (and waiting for second number)
'4 is displaying answer
'5 is moving onto next digit

DEBUG "starting program"
```

```
SEROUT 15, 84, [22, 12] ' Initialize LCD
PAUSE 5 ' 5 ms delay for clearing display

PAUSE 5 ' 5 ms delay for clearing display

SEROUT 15, 84, [128, "          "]
SEROUT 15, 84, [148, "          "]

'--------------------------------------------------------------------------------
---------------------------
  ObtainFingerPos:
    HIGH 1                                ' charge the cap
    PAUSE 5                               '   for 1 ms
    RCTIME 1, 1, fing1                   ' measure RC discharge time
    IF fing1<500 THEN                    'THRESHOLD CONTROL for finger 1
      binaryfinger1 = 1
    ELSE
      binaryfinger1 = 0
    ENDIF

    HIGH 2                                ' charge the cap
    PAUSE 5                               '   for 1 ms
    RCTIME 2, 1, fing2                   ' measure RC discharge time
    IF fing2 < 1000 THEN
      binaryfinger2=1
    ELSE
      binaryfinger2 = 0
    ENDIF

    HIGH 3                                ' charge the cap
    PAUSE 5                               '   for 1 ms
    RCTIME 3, 1, fing3                   ' measure RC discharge time
    IF fing3 < 1000 THEN
      binaryfinger3 = 1
    ELSE
      binaryfinger3 = 0
    ENDIF

    HIGH 4                                ' charge the cap
    PAUSE 5                               '   for 1 ms
    RCTIME 4, 1, fing4                   ' measure RC discharge time
    IF fing4 < 1000 THEN
      binaryfinger4 = 1
    ELSE
      binaryfinger4 = 0
    ENDIF

    HIGH 5                                ' charge the cap
    PAUSE 5                               '   for 1 ms
```

```
    RCTIME 5, 1, fing5                   ' measure RC discharge time
    IF fing5<3000 THEN
      binaryfinger5 = 1
    ELSE
      binaryfinger5 = 0
    ENDIF



    GOTO ObtainButtonStates
'--------------------------------------------------------------------------------
-------------------------------------
  ObtainButtonStates:
    Butt1 = IN6
    Butt2 = IN7
    Butt3 = IN8
    Butt4 = IN9
    GOTO ObtainNumber
'--------------------------------------------------------------------------------
-------------------------------------
  ObtainNumber:
    IF binaryfinger5 = 0 THEN
      IF binaryfinger2 = 0 THEN
        aslnumber=5
      ELSE
        aslnumber=3
      ENDIF
    ELSE
      IF binaryfinger4 = 0 THEN
        IF binaryfinger3 = 0 THEN
          IF binaryfinger2 = 0 THEN
            IF binaryfinger1 = 0 THEN
              aslnumber = 4
            ELSE
              aslnumber = 6
            ENDIF
          ELSE
            IF binaryfinger1 = 0 THEN
              aslnumber = 7
            ELSE
              aslnumber = 2
            ENDIF
          ENDIF
        ELSE
          IF binaryfinger2 = 0 THEN
            aslnumber = 8
          ELSE
            aslnumber = 1
          ENDIF
```

```
        ENDIF
      ELSE
        IF binaryfinger3 = 0 THEN
          aslnumber=9
        ELSE
          aslnumber=0
        ENDIF
      ENDIF
    ENDIF
    FOR idx = 1 TO 8
      PULSOUT 13, 750+(10*aslnumber)
      PAUSE 20
    NEXT
    GOTO StateMachine
'--------------------------------------------------------------------------------
----------------------------------------
'New Constrolstate
'1 is waiting for first digit of num
'2 is waiting for next digit of num
'3 is output

'Butt1 VAR Bit 'add
'Butt2 VAR Bit 'subtract
'Butt3 VAR Bit 'equal
'Butt4 VAR Bit 'next digit
'controlstate = 1 at default

'operator
'1 is add
'2 is subtract
'3 is multiply
'4 is divide

  StateMachine:
    SEROUT 15, 84, [17] ' Initialize LCD
    IF IN10 = 0 THEN
      DEBUG "killswitch activated"
      END
    ENDIF

'----------------------------------------------------------------
    IF controlstate = 1 THEN  'We waiting for first digit
      SEROUT 15, 84, [128+b, " "]
      PAUSE 100
      SEROUT 15, 84, [128+b, DEC aslnumber]

      digi1store =  aslnumber
```

```
IF Butt1 = 0 THEN 'add
  SEROUT 15, 84, [132, "+"]



  op = 1

  PAUSE 1000

  Butt1 = IN6
  IF Butt1 = 0 THEN
    SEROUT 15, 84, [132, "*"]
    op = 3
  ENDIF


  IF counter = 0 THEN
    diginstore = diginstore + digi1store
  ELSEIF counter = 1 THEN
    diginstore = diginstore/10 + digi1store
  ELSEIF counter = 2 THEN
    diginstore = diginstore/100 + digi1store
  ELSEIF counter = 3 THEN
    diginstore =  digi1store

  ENDIF

  digi1store = diginstore
  SEROUT 15, 84, [128, DEC digi1store]

  controlstate = 2
  diginstore = 0
  digi2store = 0
  counter = 3
  b = 0

ELSEIF Butt2 = 0 THEN 'go to subtract

  SEROUT 15, 84, [132, "-"]
  op = 2
  PAUSE 1000
  Butt2 = IN7
  IF Butt2 = 0 THEN
    SEROUT 15, 84, [132, "/"]
    op = 4
    digi1store = digi1store

  ENDIF
```

```
    IF counter = 0 THEN
      diginstore = diginstore + digi1store
    ELSEIF counter = 1 THEN
      diginstore = diginstore/10 + digi1store
    ELSEIF counter = 2 THEN
      diginstore = diginstore/100 + digi1store
    ELSEIF counter = 3 THEN
      diginstore =  digi1store

    ENDIF

    digi1store = diginstore
    SEROUT 15, 84, [128, DEC digi1store]



    controlstate = 2
    diginstore = 0
    digi2store = 0
    counter = 3
    b = 0

  Butt4 = IN9
  ELSEIF Butt4 = 0 THEN  'go to next digit

    controlstate = 1

    digi1store = aslnumber

    IF counter >=1 THEN
      FOR idx = 1 TO counter
          digi1store =  digi1store*n


      NEXT
    ENDIF
    IF  b = 3 THEN
      b = 3
    ELSE
      b = b+1
    ENDIF
    IF counter = 0 THEN
      counter = 0
    ELSE
      counter = counter -1

    ENDIF
```

```
    diginstore = diginstore + digi1store



  ENDIF
'-----------------------------------------------------------------

ELSEIF controlstate = 2 THEN 'is waiting for next digit of num
    SEROUT 15, 84, [133+b, " "]
    PAUSE 100
    SEROUT 15, 84, [133+b, DEC aslnumber]
    digi2store = aslnumber

  IF Butt3 = 0 THEN
    controlstate = 3

    IF counter = 0 THEN
      diginstore = diginstore + digi2store
    ELSEIF counter = 1 THEN
      diginstore = diginstore/10 + digi2store
    ELSEIF counter = 2 THEN
      diginstore = diginstore/100 + digi2store
    ELSEIF counter = 3 THEN
      diginstore = digi2store
    ENDIF

    digi2store = diginstore
    SEROUT 15, 84, [133, DEC digi2store]
  ENDIF

  Butt4 = IN9
  IF Butt4 = 0 THEN  'go to next digit
    controlstate = 2


    digi2store = aslnumber
    IF counter >=1 THEN
      FOR idx = 1 TO counter
          digi2store =  digi2store*n

      NEXT
    ENDIF
    IF  b = 3 THEN
      b = 3
    ELSE
      b = b+1
    ENDIF
    IF counter = 0 THEN
```

```
        counter = 0
      ELSE
        counter = counter -1
      ENDIF

      diginstore = diginstore + digi2store

      DEBUG DEC ? diginstore

    ENDIF

  ELSEIF controlstate = 3 THEN
     DEBUG DEC ? op
     SEROUT 15, 84, [148, "="]
     IF op = 1 THEN
      finalmathanswer = digi1store + digi2store
      PAUSE 1000
      SEROUT 15, 84, [149, DEC finalmathanswer]
     ELSEIF op = 2 THEN
      finalmathanswer = digi1store - digi2store

      PAUSE 1000
      IF digi1store >= digi2store THEN
        SEROUT 15, 84, [149, DEC finalmathanswer]
      ELSE
        SEROUT 15, 84, [149, "-"]
        SEROUT 15, 84, [150, DEC (65535+1 -finalmathanswer)]
      ENDIF
     ELSEIF op = 3 THEN
      finalr = digi1store**digi2store
      IF finalr = 0 THEN
        finalmathanswer = digi1store * digi2store
        SEROUT 15, 84, [149, DEC finalmathanswer]
      ELSE
        SEROUT 15, 84, [149, "OVERFLOW"]

      ENDIF



      SEROUT 15, 84, [149, DEC finalmathanswer]
    ELSEIF op = 4 THEN
      finalmathanswer = digi1store / digi2store
      finalr = digi1store//digi2store

      temp = (finalr*1000)/digi2store

      PAUSE 1000
      SEROUT 15, 84, [149, DEC finalmathanswer]
```

```
    SEROUT 15, 84, [151, "."]
    SEROUT 15, 84, [152, DEC temp]

   ENDIF
   controlstate = 4
 ELSEIF controlstate = 4 THEN
    PAUSE 5000
    SEROUT 15, 84, [128, "          "]
    SEROUT 15, 84, [148, "          "]
    controlstate = 1
    op = 1
    digi1store = 0
    diginstore = 0
    digi2store = 0
    counter = 3
    b = 0
    controlstate = 1

ENDIF

GOTO ObtainFingerPos
```