Accelerating Neural Network Inference Computation with Resistive Devices

ECE4012 Senior Design Project

Section L4A, Resistive NN Emulator Team
Project Advisor: Dr. Shimeng Yu

Team Members:
Yunfeng Xin (yxin34@gatech.edu)
Sho Ko (sko.45@gatech.edu)
Runfeng Chen (rchen324@gatech.edu)
William Scott (wrscott@gatech.edu)
William Trimmer (wtrimmer3@gatech.edu)
Zheyuan Xu (zxu322@gatech.edu)

Submitted

2019 May 2

Executive Summary

Neural networks are the foundation of modern image classification tasks. However, a typical neural network is computation-demanding and time-consuming in digital data pipelines such as Central Processing Units (CPU) and Graphics Processing Units (GPU). In order to expedite the computation, a novel accelerator that can perform efficient floating-point multiplication operation is needed. This project designs a hardware emulating system for neural network computation that took advantage of Ohm's law and Kirchhoff's voltage law, which describe relationships between voltage, resistance, and current similar to floating point addition and multiplication operation, to accelerate the computation process. The system includes a resistor network that connects input ports and output ports with a series of resistors. The resistor network takes in image pixel data that are mapped to different voltage values and outputs current levels representing different calculation results. The system also includes a microprocessor to preprocess the image to be fed into the resistor network and show the final results of the resistor network computation. Since the entire system eliminates the need for time-consuming digital adders and multipliers as required in traditional CPUs and GPUs, it reduces the time needed for neural network computation. The entire system is also considered cost-effective as both major components --- resistors and a simple microcontroller --- cost no more than $20 when manufactured in great quantities. The system in this project is capable of recognizing handwritten digits, but it can be applied to other tasks as well with minor modification.

Table of Contents

Accelerating Neural Network Inference Computation with Resistive Devices

1. Introduction

1.1 Objective

The team has designed and prototyped a resistor network system that emulates the process of matrix multiplication operations that are essential to neural network computation. A microcontroller unit (MCU) takes in images of handwritten digits, convert the pixel data into corresponding two voltage levels, and feed them into the resistor network. The resistor network, which consists of arrays of resistors with different resistance representing the weights of the nodes, converts the voltage levels into different current levels. The microcontroller then reads the current output of the resistor network and determines which digits the original inputs represent.

1.2 Motivation

Most of the neural network computation tasks are carried on digital data pipelines such as CPUs and GPUs and rely on digital multipliers to perform floating point multiplication [1]. A digital multiplier unit typically takes three nanoseconds to complete such an operation, which leaves its processor unit idle for more than 10 cycles [2]. By eliminating the need for such a digital arithmetic unit, floating point operations can be less time-consuming.

Modern CPU prices range from $100 to $1000, while modern GPU prices range from $200 to $3000 [3]. However, the team anticipates that a design without such a multiplier will potentially accelerate the computation. An emulator system with resistive networks is more cost-effective when compared to digital multipliers because one floating point multiplication requires only one resistor in the proposed system, while a digital multiplier consists of more than 20,000 transistors [2].

Similar ideas exist in the fields of memory technology such as Resistive Random-Access Memories (RRAM), but most of the concepts are still under research and there are currently no similar commercial products [4]. The product can be used by users that require efficient processing of images in time-sensitive tasks such as real-time handwritten zip code recognition on envelopes and increase the throughput of the tasks.

1.3 Background

There has been extensive research in accelerating neural network computation with a resistive network. The idea was first proposed in 2016 for multi-layer perceptron (MLP) neural network [5]. In the past two years, similar resistive accelerator solutions have been proposed for convolutional neural networks (CNN) and recurrent neural networks (RNN) as well [6], [7]. These techniques mainly focus on semiconductor fabrication level application. Due to the non-linear nature of the fabricated devices, however, no commercial devices are publicly available [5].

There are two key building blocks in this area: applying Ohm's law that resembles floating point multiplication and applying Kirchhoff's current law that resembles floating point addition. Ohm's law states that the voltage across a resistor is determined by the current going through the resistor multiplied by the resistance of the resistor. This law makes performing analog floating-point multiplication possible by providing the input voltage level and resistance values and measuring the output current value as the result of the multiplication. Kirchhoff's current law states and the total current output is equal to the sum of all input current. This law makes performing analog floating-point addition possible by providing two analog current values and measuring the output current value as the result of the addition.

2. Project Description and Goals

The fundamental goal of the resistive neural network emulator system is to build a proof-of-concept system showing that resistor networks can act as hardware accelerators for performing neural network algorithms. Our hardware system consists of a MCU, a resistor network, and three on-board ADCs which work together to represent the network data as analog voltage and current values. For the purpose of reducing hardware complexity, the MCU in our system is used for both image preprocessing and computation of other layers in the neural network architecture. The image preprocessing functions running on MCU includes normalization and re-scaling. Since the system emulates only one layer of fully-connected neurons, computation of other layers are performed on the MCU. The resistor network system contains more than 60 resistors, with around 20 resistors per column and 3 columns in total. Each network column's current reflects the confidence level of the

model's classification on the corresponding number, which is translated back to floating number through an ADC and feedback to the MCU. The MCU takes the numbers and displays the classification result on a GUI. The features of our system include:

- Accelerated hardware-based handwritten image recognition
- Cost-effective design with only one MCU in total
- Customizable neural network configuration through changing resistors
- GUI for displaying classification result

3. Technical Specifications

3.1 Resistive System Specification

Table 1. Neural Network model specification

Item	**Specification**
Detection Accuracy	98.67%
Execution Time	70 ns
Output Nodes	3
Input Nodes	64
Hidden Layer Number	1

3.2 Hardware System Specification

Table 2. Resistor Network specification

Item	Specification
Quantization Level	2
Resistor Range	0 Ω or 30 kΩ
Resistor Mounting	Through Hole

Table 3. MCU Specification

Item	Specification
Analog Input Pin Number	10
Analog Output Pin Number	64
Power Supply	5 V
ADC Resolution	8 bits
Interfaces	USB Serial
RAM	512 KB

Table 4. ADC Specification

Item	Specification
Resolution	8 bits
Voltage Sense Range	0 - 3.3V
Interfaces	N/A (MCU on-board)

Supply Voltage	N/A (MCU on-board)

4. Design Approach and Details

4.1 Design Approach

In the resistive network layer, hardware components are used to implement the weights in a single layer of a neural network. For this design, analog voltages are passed into the matrix of resistors, producing analog output currents representing the weighted information values. To implement this, the design uses an STM32F767ZI Mbed-compatible microcontroller to handle file I/O, 3 on-board ADCs to measure the output currents, and 64 on-board digital out pins to supply the input voltages.

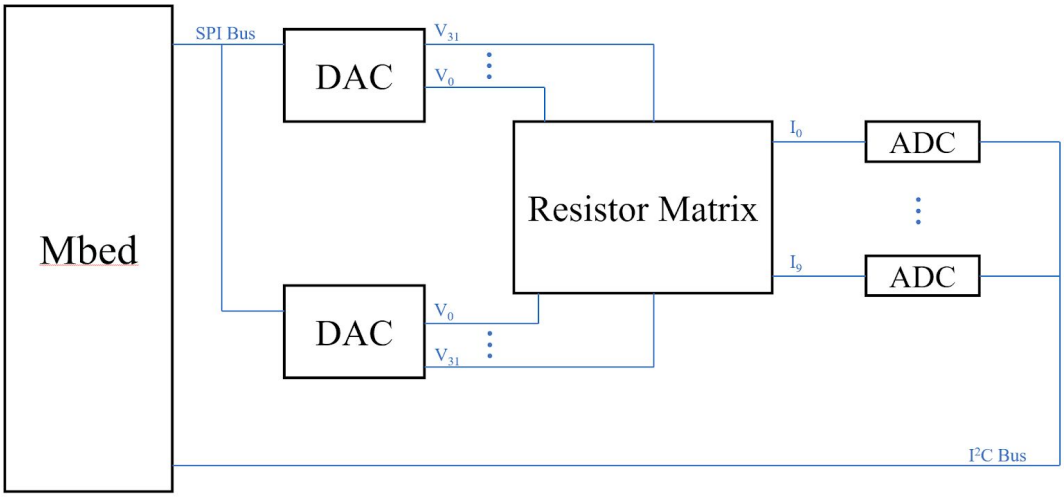


Figure 1. Block diagram of the resistive neural network emulator system.

4.1.1 Microcontroller

The Mbed-compatible STM32F767ZI MCU is used to read inputs and control output voltages. This module was chosen since the students have previous experience programming with ARM Mbed interface, and it has sufficient capabilities to perform the necessary I/O and processing tasks.

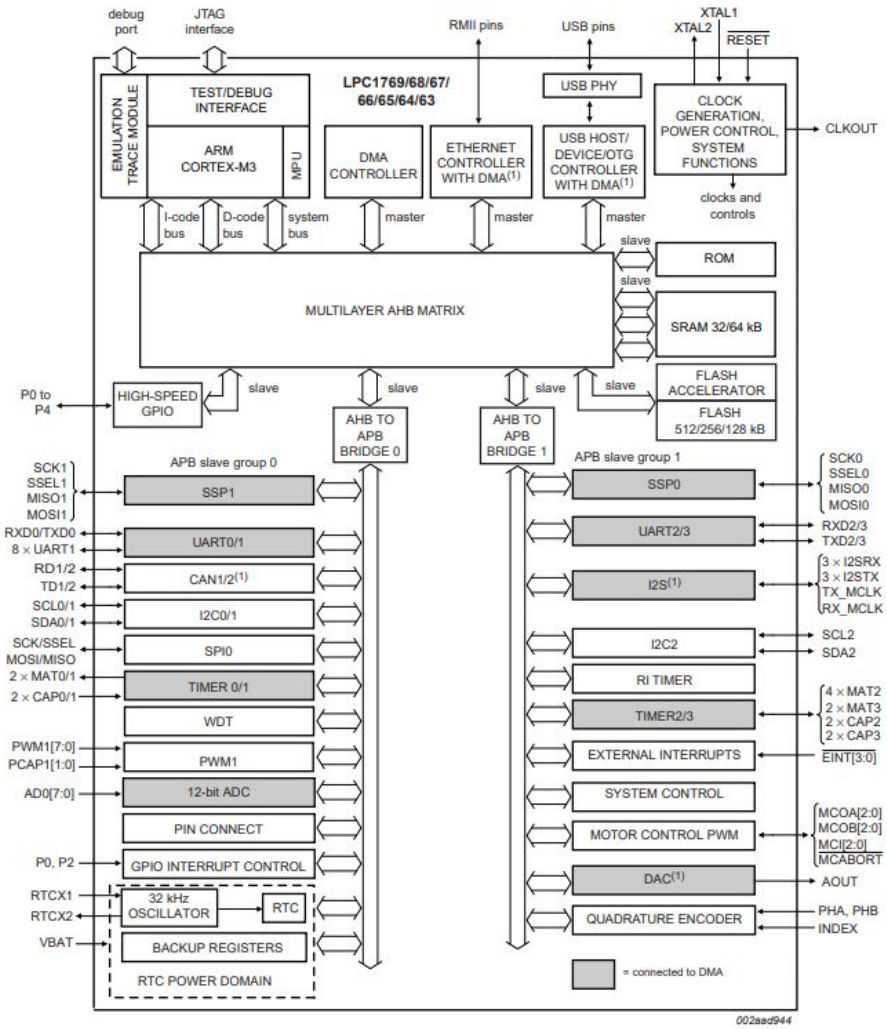


Figure 2. Block diagram of the NXP Semiconductor Mbed LCP1768 microcontroller.

4.1.2 Operational Amplifier (Op-Amp)

Three TL084CN operational amplifier chips from Texas Instruments [10], along with some sensing resistors, converts the output current values to readable voltages for the three ADCs. This avoids having the ADCs placed in series with the output currents, which would change the overall resistance of the network. These chips were chosen for their cheap pricing.

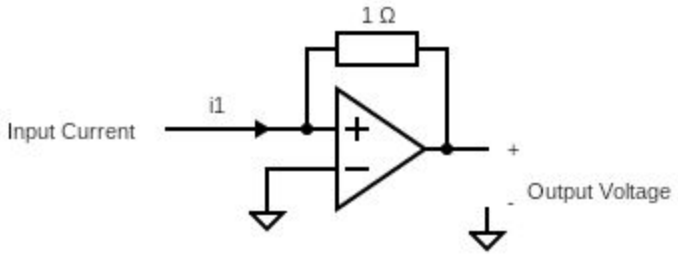


Figure 4. Circuit diagram for converting current value into voltage values.

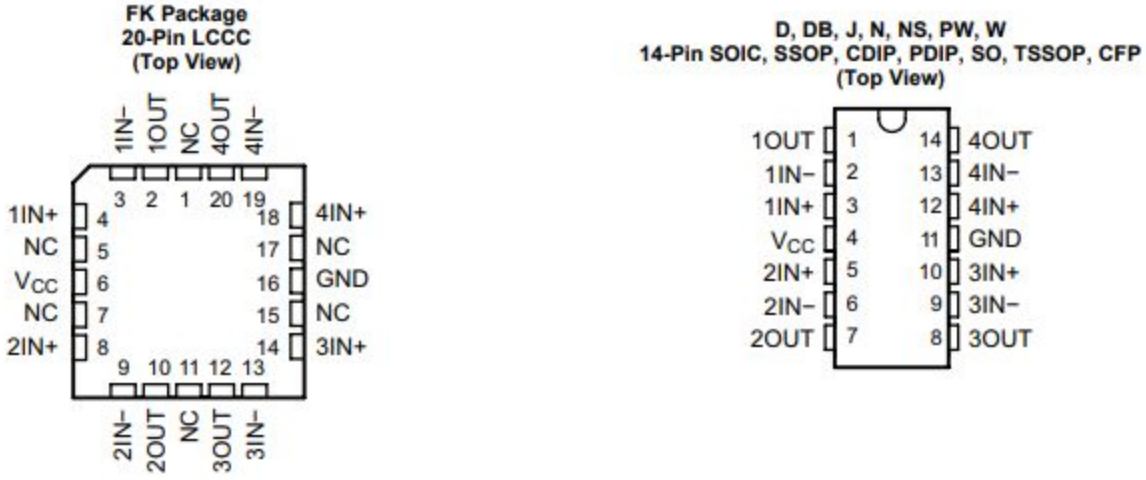


Figure 5. Pinout diagram of Texas Instruments LM324ADR operational amplifier chip.

4.2 Standards and Constraints

The overarching constraint on the design is having each component communicate effectively with the microcontroller. Fortunately, the Mbed microcontroller supports many communication standards (SPI, I^2C, Serial, CAN, Ethernet, USB), thus the design can utilize many unique chipsets. Another constraint is managing the currents through the resistor matrix. Each digital out pin can only handle a certain load resistance, as well as a maximum current output. The values for the resistors are scaled to conform to these thresholds. Cost also plays a significant role in the design, as it requires many components that tend to vary in price. The on-board ADCs are extremely cost-effective considering the fact that they meet the design specifications and do not place extra cost to the design. When searching for resistors, the team also had to trade accurate resistor values for cheaper resistors.

5. Schedule, Tasks, and Milestones

The Resistive NN Emulator team designed and implemented this prototype in Spring 2019. Appendix A contains the task table and the corresponding Gantt chart outlining the time of major tasks. Appendix B contains a more detailed task table with estimated best and worst time scenarios, risk analysis, and team members assigned to the tasks. Appendix C provides a PERT chart with tasks, milestones, and critical paths shown.

6. Project Demonstration

The system used for the testing process of our project is considered to be portable with multi-platform capabilities. The general testing procedure was conducted by one or more people doing the following:

1. The MCU is programmed with pre-trained machine-learning algorithms.
2. The MCU is connected to the resistive network—consisting of more than 60 commercially available resistors—by on-board digital out pins, where it sends the analog voltage values.
3. The MCU generates inputs and feeds them into the resistive network. In our case, the inputs are an encoded matrix of data containing the information of an 8-pixel by 8-pixel image file.
4. After passing through the trained resistive network, the data is read by the MCU from the on-board ADCs connected to the network. The output data is collected and cached on the PC. The data is visualized and analyzed by the user and compared to the target output to evaluate the performance.

For better visualization, a PC is connected to the setup:

- **Output data acquisition**: The output data can be acquired directly from the MCU by writing to a serial console on a PC. This stores the results of the network classification to be used for later information processing.
- **Data visualization**: The data collected are visualized on a webpage. For real-time visualization, a serial console to a PC is used to communicate current status to the user.
- **Output comparison**: The collected data are visualized and compared with the target output. The difference can be used to analyze the performance of the resistive network.

- **Samples**: Due to the small size of a single sample image, multiple images from the data set or even user-generated images are fed into the network to provide a more thorough analysis of the performance.

7. Marketing and Cost Analysis

7.1 Marketing Analysis

Most deep learning algorithms like neural networks run on digital processors such as CPUs and GPUs, which are power-hungry. Researchers are shifting towards hardware-based neural networks, such as in resistive processing units (RPUs), for speedup in the training process and reduction in power consumption. Although most RPUs are still in the phase of academic research and not commercially available, RPUs have great market potential once they are manufactured and targeted at the appropriate clients.

Compared to CPUs and GPUs, the RPUs have advantages in ways of exploiting analog components of a circuit to simulate a neural network. Specifically, the synaptic weights of the neural network can be represented by the conductance of the resistors, while in digital systems, the digitization process slows down the computation time and causes loss of accuracy. Therefore, this can be the first marketing advantage for RPUs over CPUs and GPUs. In addition, digital processors are good for general-purpose computation but are not computationally efficient at some specific neural networks, while RPUs can be application-specific in the sensor that they have a narrower range of application domain but are very powerful at the tasks in that specific domain. Thus, this makes the second marketing advantage for RPUs over CPUs and GPUs.

7.2 Cost Analysis

Assume there is a startup company selling RPU chips. The company hires engineers to do designing which includes system specification, architectural design, functional design, logic design, and circuit design. Assume the company hires around 40 college graduates with bachelor's degree in either electrical or computer engineering and pay them with starting salary for engineers around $80,000 per year. The annual payment of the company is around $3,000,000. The project of designing RPU chips lasts for two year and each engineer spends ten hours in the first year working on circuit and chip design. The company also needs supplies such as power supplies and cables, which cost around $1,000,000 per year. After the design process, the company sends the design layout to another company for fabrication and packaging [11]. It usually takes around several months for the fabrication process and the cost is around $4,000,000 [12]. After the chips are fabricated and sent back, the company needs to assemble and test the units. This will take several more months in the second for the engineers to finish their work. Additionally, we also take into consideration factors such as fringe benefits, overheads and sale expenses, which cost another $1,000,000. The total cost in the two years of design, fabrication, testing, and overheads is around $12,000,000. Assume we sell the RPU chips in the period of five years and in each year, we sell around 50 thousand chips. We amortize the development costs over all this units and we get around 48 dollars. Assume we sell the chip at around 70 dollars, which is lower than most CPU prices which range from $100 to $1000 and GPU prices which range from $200 to $3000 [3]. The expected profit for each chip is around $22 and the percent profit is around 45%.

8. Conclusion

In conclusion, our team has successfully designed a resistive system to emulate the computation of convolutional neural network with the help from a MCU. We are able to achieve 98.67% accuracy on handwritten digit recognition tasks. In addition, we significantly reduce the time and energy consumption of the computation by a magnitude of a thousand. The system can be further improved in the future by extending the resistor network to include the convolution layer on the hardware side. The whole system can also be migrated to a printed circuit board to reduce the total area.

9. References

[1] S. Yalamanchili, ECE 8823 GPU Architecture. [Online]. Available: http://ece8823-sy.ece.gatech.edu/. [Accessed: Oct. 21, 2018].

[2] M. Olivieri, “Design of synchronous and asynchronous variable-latency pipelined multipliers,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 9, no. 2, pp. 365–376, 2001.

[3] J. Hruska, “Charting 9 Years of GPU Market Shifts Between Intel, AMD, and Nvidia,” ExtremeTech, 05-Sep-2018. [Online]. Available: https://www.extremetech.com/gaming/276425-charting-9-years-of-gpu-market-shifts-between-intel-amd-and-nvidia. [Accessed: Nov. 28, 2018].

[4] S. Yu, "Neuro-inspired computing with emerging nonvolatile memorys," in Proceedings of the IEEE, vol. 106, no. 2, pp. 260-285, Feb. 2018.

[5] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, “ISAAC: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars,” 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), pp. 1–4, 2016

[6] T. Gokmen, M. J. Rasch, and W. Haensch, “Training LSTM Networks With Resistive Cross-Point Devices,” Frontiers in Neuroscience, vol. 12, p. 1, 2018.

[7] T. Gokmen and Y. Vlasov, “Acceleration of Deep Neural Network Training with Resistive Cross-Point Devices: Design Considerations,” Frontiers in Neuroscience, vol. 10, p. 1, 2016.

[8] Analog Devices, “32-Channel, 16-/14-Bit, Serial Input, Voltage Output DAC”, AD5372 datasheet, Nov. 2011.

[9] Microchip, “High-Side Power/Current Monitor with Analog Output”, PAC1921 datasheet, 2016.

[10] Texas Instruments, “Quadruple Operational Amplifier”, LM324ADR datasheet, 2018.

[11] V. Mooney, ECE 8873 Advanced Hardware-Oriented Security and Trust. [Online]. Available: http://mooney.gatech.edu/Courses/ECE4823/index.html. [Accessed: Nov. 27, 2018].

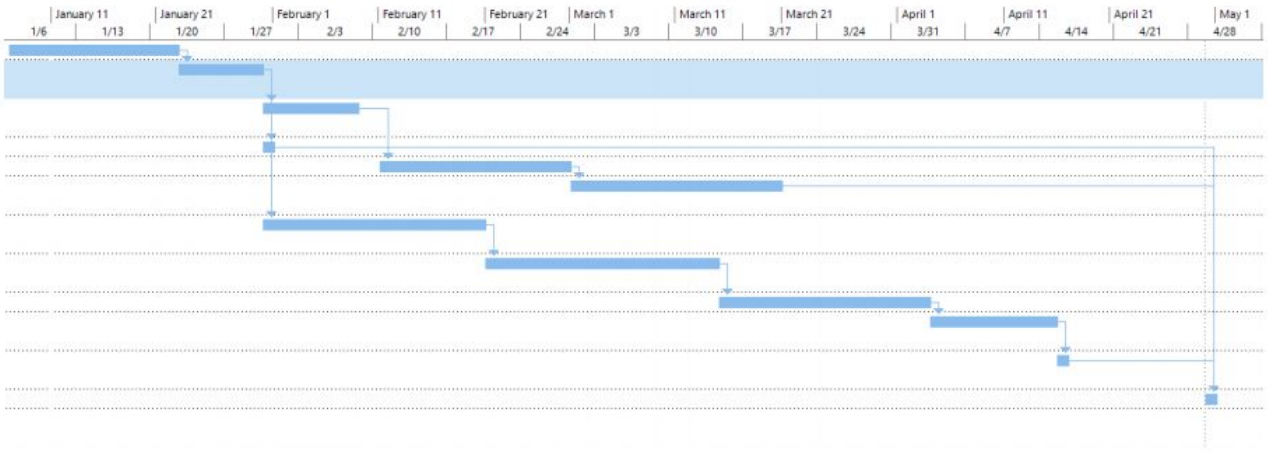
[12] G. Burton, “TSMC says 3nm plant could cost it more than $20bn,” Oct, 2017. [Online]. Available: https://www.theinquirer.net/inquirer/news/3018890. [Accessed: Nov. 27, 2018].

Appendix A - Task Table and Gantt Chart

Task Table

	Task Name	Duration	Start	Finish	Predecessors
1	Build and train CNN	12 days	Mon 1/7/19 8:00 AM	Tue 1/22/19 5:00 PM	
2	Extract weights from CNN	6 days	Wed 1/23/19 8:00 AM	Wed 1/30/19 5:00 PM	1
3	Determine accuracy and quantization	7 days	Thu 1/31/19 8:00 AM	Fri 2/8/19 5:00 PM	2
4	Oral presentation	1 day	Thu 1/31/19 8:00 AM	Thu 1/31/19 5:00 PM	2
5	Buy resistors	14 days	Mon 2/11/19 8:00 AM	Thu 2/28/19 5:00 PM	3
6	Build resistor network on PCB	14 days	Fri 3/1/19 8:00 AM	Wed 3/20/19 5:00 PM	5
7	Implement ADC and input control logic	15 days	Thu 1/31/19 8:00 AM	Wed 2/20/19 5:00 PM	2
8	Implement DAC and output control logic	16 days	Thu 2/21/19 8:00 AM	Thu 3/14/19 5:00 PM	7
9	User interface	14 days	Fri 3/15/19 8:00 AM	Wed 4/3/19 5:00 PM	8
10	System integration and debugging	8 days	Thu 4/4/19 8:00 AM	Mon 4/15/19 5:00 PM	9
11	Final project demonstration	1 day	Tue 4/16/19 8:00 AM	Tue 4/16/19 5:00 PM	10
12	Design Expo	1 day	Tue 4/30/19 8:00 AM	Tue 4/30/19 5:00 PM	11,4,6

Gantt Chart



Appendix B - Detailed Task Table

	Task Name	Duration	Start	Finish	Predecessors	Best	Worst	Expected	Standard Deviation	Team Member	Risk
1	Build and train CNN	12 days	Mon 1/7/19 8:00 AM	Tue 1/22/19 5:00 PM		10 days	14 days	12 days	0.67 days	All	Medium
2	Extract weights from CNN	6 days	Wed 1/23/19 8:00 AM	Wed 1/30/19 5:00 PM	1	4 days	10 days	6.33 days	1 day	All	High
3	Determine accuracy and quantization	7 days	Thu 1/31/19 8:00 AM	Fri 2/8/19 5:00 PM	2	4 days	10 days	7 days	1 day	All	High
4	Oral presentation	1 day	Thu 1/31/19 8:00 AM	Thu 1/31/19 5:00 PM	2	1 day	1 day	1 day	0 days	All	Low
5	Buy resistors	14 days	Mon 2/11/19 8:00 AM	Thu 2/28/19 5:00 PM	3	11 days	18 days	14.17 days	1.17 days	Yunfeng Xin, Rungfeng Chen	Medium
6	Build resistor network on PCB	14 days	Fri 3/1/19 8:00 AM	Wed 3/20/19 5:00 PM	5	10 days	17 days	13.83 days	1.17 days	William Scott, William Trimmer, Zheyuan Xu, Sho Ko	High
7	Implement ADC and input control logic	15 days	Thu 1/31/19 8:00 AM	Wed 2/20/19 5:00 PM	2	12 days	19 days	15.17 days	1.17 days	Yunfeng Xin, Runfeng Chen, Sho Ko	High
8	Implement DAC and output control logic	16 days	Thu 2/21/19 8:00 AM	Thu 3/14/19 5:00 PM	7	14 days	21 days	16.5 days	1.17 days	William Scott, William Trimmer, Zheyuan Xu	High
9	User interface	14 days	Fri 3/15/19 8:00 AM	Wed 4/3/19 5:00 PM	8	11 days	19 days	14.33 days	1.33 days	All	Low
10	System integration and debugging	8 days	Thu 4/4/19 8:00 AM	Mon 4/15/19 5:00 PM	9	5 days	11 days	8 days	1 day	All	High
11	Final project demonstration	1 day	Tue 4/16/19 8:00 AM	Tue 4/16/19 5:00 PM	10	1 day	1 day	1 day	0 days	All	Medium
12	Design Expo	1 day	Tue 4/30/19 8:00 AM	Tue 4/30/19 5:00 PM	11,4,6	1 day	1 day	1 day	0 days	All	Medium

Appendix C - Project Pert Chart