

Universal Circuit Fabricator (UCF)

Project Documentation

EECS Senior Design Group 18 2014-2015

Erolle Dayuta

Hector Melendez

Kyle Scott



UNIVERSITY OF CENTRAL FLORIDA

Table of Contents

1	Executive Summary	1
2	Project Description	2
2.1	PERSONNEL	2
2.2	PROJECT MOTIVATION	2
2.3	GOALS & OBJECTIVES	3
2.3.1	System Goals.....	4
2.4	REQUIREMENTS & SPECIFICATIONS	5
2.4.1	Overall System.....	5
2.4.2	Printing Process	5
2.4.3	Data Transmission	5
3	Research	6
3.1	RELEVANT TECHNOLOGIES	6
3.1.1	Conductive Ink Choices.....	6
3.1.2	Conductive Ink Decision	10
3.1.3	Printing Substrate Choices	10
3.1.4	Printing Substrate Decision	12
3.2	EXISTING PROJECTS & PRODUCTS	12
3.2.1	Transit Tracking System.....	12
3.2.2	AgIC Printer Kickstarter Design	14
3.2.3	The EX Rapid 3D Circuit Printer	15
4	Project Architecture Selection	17
4.1	POSSIBLE ARCHITECTURES CHOICES	17
4.1.1	Possible Hardware Architecture Choices.....	17
4.1.2	Possible Software Architecture Choices	19
4.2	SELECTION OF INITIAL DESIGN ARCHITECTURE	20
4.2.1	Hardware Architecture Selection	21
4.2.2	Software Architecture Selection.....	21
4.3	OVERALL SYSTEM ARCHITECTURE	21
5	Prototype Hardware Design	23
5.1	SYSTEM LEVEL DESIGN.....	23
5.1.1	Component Details & Configuration.....	23
5.1.2	Block Diagram.....	24
5.2	INKJET CARTRIDGE CONTROL SYSTEM.....	24
5.2.1	Component Details & Configuration.....	24

5.2.2	Block Diagram.....	25
5.2.3	InkShield Parts List.....	26
5.3	MOTOR CONTROL SYSTEM.....	26
5.3.1	Component Detail & Configuration	27
5.3.2	Block Diagram.....	27
5.4	ANNEALING SYSTEM.....	28
5.4.1	Component Detail & Configuration	28
5.4.2	Block Diagram.....	28
5.5	DATA INPUT & PROCESSING SYSTEM.....	29
5.5.1	Component Detail & Configuration	29
5.5.2	Block Diagram.....	30
5.6	MICROCONTROLLER SYSTEM	31
5.6.1	Component Details & Configuration.....	31
5.6.2	Block Diagram.....	33
6	Software Design	34
6.1	SYSTEM LEVEL DESIGN.....	34
6.1.1	Main Function Description	34
6.2	MICRO-CONTROLLER CODE	35
6.2.1	Variable and Pin Descriptions.....	36
6.3	INKJET CARTRIDGE CONTROL SYSTEM.....	37
6.3.1	Software Configuration.....	37
6.3.2	Variable Descriptions	38
6.3.3	Function Descriptions.....	39
6.4	MOTOR CONTROL SYSTEM.....	39
6.4.1	Software Configuration.....	39
6.4.2	Variable Descriptions	41
6.4.3	Function Descriptions.....	42
6.5	ANNEALING SYSTEM.....	43
6.5.1	Software Configuration.....	43
6.5.2	Variable Descriptions	43
6.5.3	Function Descriptions.....	44
6.6	DATA COLLECTION & PROCESSING SYSTEM.....	44
6.6.1	Software Configuration.....	44
6.6.2	Variable Descriptions	46
6.6.3	Function Descriptions.....	47
6.7	GRAPHICAL USER INTERFACE	48

6.7.1	Software Detail & Configuration.....	48
6.7.2	GUI Code Composer.....	49
6.7.3	Capacitive Touch Input.....	50
6.7.4	LCD Display	51
6.7.5	Block Diagram.....	52
7	Prototype Design Summary	53
7.1	INPUT SUBSYSTEMS.....	53
7.2	PROCESSING SUBSYSTEM	53
7.3	PRINTING SUBSYSTEMS.....	53
7.4	OUTPUT SUBSYSTEM	53
8	Build Plan	54
8.1	PROTOTYPE DESIGN	54
8.1.1	Hardware Development.....	55
8.1.2	Software Development	64
8.2	FINAL DESIGN	66
8.2.1	Hardware Final Design Differences	66
8.2.2	Software Final Design Differences	67
8.2.3	PCB Layout Design	67
8.2.4	PCB Parts List.....	70
8.2.5	Final Design Housing and Frame	71
9	Design Constraints and Standards	72
9.1	REALISTIC DESIGN CONSTRAINTS	72
9.1.1	Economic Factors.....	72
9.1.2	Manufacturability	72
9.1.3	Health and Safety.....	72
9.1.4	Reliability.....	72
9.1.5	Environmental Impact.....	73
9.1.6	Ethical	73
9.2	STANDARDS	74
10	User Manual	74
10.1	HARDWARE SETUP	74
10.1.1	USB Serial Interface.....	75
10.1.2	Stepper Motor Interface.....	75
10.1.3	InkShield Interface.....	75
10.1.4	Hardware Switch Interface	76

10.2	SOFTWARE SETUP	77
10.2.1	Circuit Trace Generation	77
10.2.2	Conversion to G-Code.....	77
10.2.3	G-Code Command Transmission	78
11	Test Plan	79
11.1	HARDWARE TESTING.....	79
11.1.1	Data Input and Processing Hardware	79
11.1.2	Motor Control Hardware	80
11.1.3	Inkjet Cartridge Control Hardware	82
11.1.4	Annealing Hardware.....	84
11.1.5	Graphical User Interface Hardware	85
11.1.6	Final Hardware System Evaluation.....	86
11.2	SOFTWARE TESTING	86
11.2.1	Data Input and Processing Software	86
11.2.2	Motor Control Software.....	88
11.2.3	Inkjet Cartridge Control System Software	90
11.2.4	Annealing System Software	91
11.2.5	Graphical User Interface Software.....	92
11.2.6	Final Software System Evaluation.....	92
11.3	OVERALL SYSTEM TESTING	93
11.3.1	Overall System Test Cases	93
11.4	OVERALL SYSTEM EVALUATION	95
12	Budget & Financing.....	96
12.1	BILL OF MATERIALS.....	96
12.2	FINANCING	99
13	Project Milestone Schedule.....	100
14	Final Project Summary / Conclusions	102
	Appendix A: References.....	A1
A.1	REFERENCES.....	A1
A.1.1	Bibliography	A1
A.2	PERMISSIONS	A2
A.2.1	Figure Reprint Permissions	A2
A.3	SCREENSHOT PERMISSIONS.....	A5
A.3.1	Arduino.....	A5
A.3.2	Energia.....	A8

1 Executive Summary

When engineers need to prototype their circuits, they utilize a tool called a breadboard. Breadboards have been used for prototyping since the 1970's. It's a very versatile tool that engineers use to prototype and experiment with their circuits without having to solder. Some of the biggest drawbacks of breadboards are cable organization, visual appeal, and their many troubleshooting challenges.

The Universal Circuit Fabricator (UCF) looks to alleviate these problems, by providing the user with a more organized circuit. Also, the Universal Circuit Fabricator allows the user to bring circuit ideas to life more efficiently than with conventional breadboards. The idea behind this project is to take a circuit diagram, made with a circuit design program, and print the circuit traces onto a specified surface with conductive ink. With this design, the user can place the corresponding components onto the printed traces so the circuit is ready for testing.

The use of this device will not only apply to engineers in the field, but it will also help engineers who are still in college. Every engineer has to take classes that will provide them with experiences to apply what they learned from class lectures. There, students have to recreate specific kinds of circuits, which are put into a circuit design program and then simulated. After that is finished, the students then have to build the physical circuit which is done utilizing a breadboard. This situation is the motivation for this group's desire to create the Universal Circuit Fabricator. The UCF will allow the user to print their schematic from an input picture, through serial input via USB with a computer. The Universal Circuit Fabricator device will approximately be the same size as conventional inkjet printers. Essentially, making it adequate for every environment it is placed in. The printing area will be 5 by 5 inches, making it a small area to work with in order to conserve ink. This design makes it appropriate for prototyping purposes, as well as learning purposes.

This document summarizes the design ideas and thought process involved in creating the Universal Circuit Fabricator. Section two shows the motivations and objectives that brought the idea to the group. In this section, there is also a list of requirements and specifications that the group must meet, in order to satisfy the project objectives. Section three seeks to demonstrate the research processes behind selecting the components and designs for the UCF. This section includes information regarding the different kinds of conductive inks, along with their properties. The group decided to test small portions of each kind of ink, to determine which had the best performance to price ratio, as well as what ink would be the best fit for the application's needs.

The latter part of section three also includes the research information gathered from other designs that have been able to achieve what this project is trying to accomplish. Some of the ideas used by these other projects will be incorporated into this design, but they will be modified to fit this project's needs. Section four

shows the project architecture designs that were taken into consideration to find the best fit for the project. This section includes a range of devices such as the Arduino UNO, and the MSP430. The group will also design the use of a Printed Circuit Board (PCB), which will be the more appropriate approach. Sections five and six show the hardware components used in the design as well as the software design utilized to operate the hardware accordingly. Sections seven, eight, and 11 are the design plan, build plan, and test plan that the group selected to develop the UCF prototype. Finally, sections 9 and 10 represent the design constraints and standards of the UCF, as well as a user manual to demonstrate how to use the project.

2 Project Description

The Project Description addresses the members of the project group, the motivation behind developing a circuit printing device, the goals and objectives of the product in development, and finally the specifications that the prototype must meet to adequately fulfill the goals of the project.

2.1 PERSONNEL

Referred to as “the group”, the following individuals developed the plan and design for the Universal Circuit Fabricator prototype. The contributing individuals used their coursework and learning experiences from their time at the University of Central Florida to create a thoughtful and in-depth design plan.

Erolle Dayuta, EE
martin.dayuta@knights.ucf.edu

Hector Melendez, EE
melendez.hector.3@knights.ucf.edu

Kyle Scott, EE
kylekscott@knights.ucf.edu

2.2 PROJECT MOTIVATION

Electrical Engineering Colleges require students to take an applications section with their respective courses, also known as a laboratory. In these laboratories, students are asked to build circuits in order to test and study the concepts behind class lectures. These laboratories allow the students to draw and test the circuit within a circuit design program. However, the current methods utilized to analyze and test discrete electrical circuits are rather disorganized and difficult to trace. Such situations make our product helpful and relevant by allowing the user to print their circuit with conductive ink on a piece of paper, a transparency or even glass. The printed circuit traces are more convenient as they are better organized and make it easier to trace the connections, when compared to a traditional breadboard's design.

A university laboratory setting is not the only application for this project. Engineers in the field could also use the Universal Circuit Fabricator to make prototypes of their circuit design. This would make the research and development departments of engineering firms work at a faster pace. The idea of being able to print circuit traces will improve circuit design dramatically. When engineers need a prototype for their project, they normally resort to utilizing a breadboard. After thorough testing and modification, they have to order a PCB in order to have a permanent design that they can work with. While ordering PCBs in bulk is cost effective, ordering small batches before the product is finalized becomes time consuming and expensive in terms of debugging; resulting in higher costs and time lost during the production and shipping of the product. The problems with PCBs are that they take quite some time to make, and once printed, cannot be altered. On a PCB, once the completed design is printed, any additions or changes to the design require the printing of a new PCB. This costs time and money that some people might not have due to the pressure of meeting deadlines. This real world issue is where the Universal Circuit Fabricator will excel, because engineers can inexpensively print multiple versions of their circuit designs throughout the testing and modifying phases. Instead of waiting until multiple modifications yield a functioning product, engineers can print a new circuit on the UCF to see if slight modifications were beneficial before moving forward. Constant monitoring with immediate feedback eliminates the need for ordering multiple PCBs, ensuring that time is best used and money is spent most cost effectively.

2.3 GOALS & OBJECTIVES

The goal of this project is to create a device that will be able to print a circuit schematic onto a piece of paper, a transparency, or even glass. This is achieved by utilizing conductive ink, and printing the circuit traces with an inkjet print head. The purpose of this project is to help the user prototype their designs without the hassle of utilizing a breadboard. It will also save time and the problems involved in troubleshooting the circuit traces that would normally be difficult to follow.

This device is also a useful tool for engineers in practice, as well as those currently in the field. It will allow them to utilize a circuit schematic that they have already created using circuit design software, by printing the previously designed circuit onto a desired surface.

One of the benefits of this product will be its ease of use. The project will be able to take an input, whether it is a picture or a circuit schematic file. The device will take two possible forms of input: a USB serial input into the microprocessor of the group's choosing, and/or an SD card input. Additionally, the group has discussed a plan to integrate a capacitive touch screen enabling the user to draw their schematic with greater ease, as well as eliminating the need for an external computer.

2.3.1 System Goals

- The UCF will operate on an Arduino UNO, only for planning and testing purposes. Once the prototype design has been proven using the prototyping board, the group will design a custom Printed Circuit Board with an Atmel ATmega328P that will work just as it did with the prototype, but with a specifically tailored embedded computer that runs the same software as the Arduino.
- The UCF will incorporate a breakout board, Inkshield, originally designed for an Arduino, as the interface between the prototyping microcontroller and the inkjet cartridge control system.
- The inkjet cartridge will be mounted onto a structure that will allow movement of the ink print head. This will be interfaced to the microcontroller by utilizing two stepper motor connected to a motor control breakout board.
- The UCF will also incorporate a serial-to-USB communication between the host PC and the PCB in order to have serial communication to transfer and receive Gcode commands.
- The UCF will receive user input through USB serial input from a computer
- The UCF will have an allotted USB port for a serial connection with a computer.
- The information will be processed by the microprocessor and then separated into two sections.
- A portion of the information will go to the InkShield board.
- The other portion of information will go to the motor control breakout board.

2.4 REQUIREMENTS & SPECIFICATIONS

In order to best demonstrate the requirements and specifications needed for a successfully operating prototype, the Universal Circuit Fabricator will be separated into several subsystems.

2.4.1 Overall System

Overall System specifications describe the numerical targets that the prototype machine must meet in order to verify a successfully functional design. System specifications will be tested and verified when the test plan is carried out.

- The UCF will have the ability to print a circuit trace in 5 minutes or less.
- It will be able to fit onto a desktop with dimensions equal to or less than four feet wide, 4 feet deep, and 4 feet tall.
- It will weigh less than or equal to 20 pounds.

2.4.2 Printing Process

Printing Process specifications describe the detailed numerical targets that the prototype machine must meet in regards to printing of conductive traces. Printing Process specifications will be tested and verified when the test plan is carried out.

- The UCF will have the ability to print continuous conductive traces with a maximum line thickness of 10 mm.
- Optimally; the line thickness will be 1 mm.
- Conductive traces will have a resistivity less than or equal to $1 \Omega \cdot \text{cm}$.
- The printing surface will allow for a printing area of 10 by 10 inches.

When the UCF receives an image file or a schematic file, it will interpret the information through a software algorithm. Then the information will be translated into instructions that the printer will follow and print the desired image.

2.4.3 Data Transmission

With the use of several breakout boards, the system's microprocessor will have to work as relay of information. Once the information is processed the microprocessor will work in conjunction with the Inkshield board, the motor control and the display control of the UCF.

- Supported file formats will include at least one of the following: Joint Photographic Experts Group File (.jpg or .jpeg), Bitmap Image File (.bmp), Portable Network Graphics File (.png)
- Optimally, the file format that the UCF will use is the bitmap (.bmp) which can be created in many third party software, like Microsoft Paint

3 Research

The Universal Circuit Fabricator relies on the concept of depositing a liquid conductive ink onto an insulating substrate via an ink jet print cartridge. Liquid conductive ink technology is a developing field that has recently gained interest in the world of electrical engineering. While there are several commercially available products that might meet the specifications of the UCF, they are expensive and proprietary. The research portion of this project will focus on the development of a homemade conductive ink that can be stored in an ink jet print cartridge and successfully print without clogging the ink jet nozzles.

In addition to research into conductive ink choices, the substrate that the ink will be printed on must also be investigated to ensure that it will be able to create the final circuit trace product that is desired through this project.

Research into previously created projects was also conducted to aid the group in developing the design, build, and test plans of the hardware and software that will make up the Universal Circuit Fabricator.

3.1 RELEVANT TECHNOLOGIES

To ensure correct functionality of the Universal Circuit Fabricator, investigation into the optimal combination of conductive ink and ink substrate is necessary.

3.1.1 Conductive Ink Choices

Research concerning the various processes employed to produce conductive ink is gathered by the group members. These ink types are produced and tested against the design specifications to identify plausible choices. The optimal ink variety is selected from the possible choices by choosing the ink with the best performance to price ratio, keeping the importance of low resistivity (Ω/cm) in mind. Some of the conductive ink configurations require annealing to transform the ink into a finalized state. Annealing is the process of heating a material and allowing it to cool down slowly in an effort to fuse the material into a continuous structure, thus toughening it and reducing resistivity.

3.1.1.1 Conductive Ink Design Requirements

- The HP C6602 inkjet cartridge must be able to store the conductive ink without leaking.
- The HP C6602 inkjet cartridge print head must be able to print a continuous line of conductive ink without clogging.

3.1.1.2 Gallium-Indium Ink

The Gallium-Indium Ink is simple to produce and yields a viscous liquid that would be free of large particulates. It is inexpensive to produce, but is expensive in material costs when compared to some of the other inks.

The alloy is comprised of 75.5% Gallium and 24.5% Indium. It is combined in a beaker of deionized water and heated to 50° Celsius to fuse the solid Indium powder into the liquid Gallium. A syringe is used to move the heavier than water ink out of the beaker and into the clean HP C6602 inkjet cartridge.

The Gallium – Indium Ink is liquid at room temperature due to the high concentration of liquid Gallium. The viscosity of the ink and its inability to cure at room temperature must be considered when testing against the design requirements. Refer to section A1.1 – Bibliography for sourcing information on the manufacturing of the ink.

3.1.1.3 Copper Sulfate Ink

The Copper Sulfate Ink is also simple to produce and yields a paint-like ink with variable viscosity that consists of copper nanoparticles. The Copper Sulfate ink is inexpensive to produce and also inexpensive in material costs. There is a longer production time because the copper nanoparticles must settle at the bottom of the mixture.

The Copper Sulfate Ink is comprised of 1 g Copper Sulfate mixed into 5 g of Ascorbic acid (Vitamin C) in 200mL of water at 70° Celsius. The reaction between the Copper Sulfate and Ascorbic acid forms Copper nanoparticles in the solution. The Copper nanoparticles are heavier than water and sink to the bottom of the solution. The precipitation of the nanoparticles takes place over a significant period of time (12-24 hours).

Once the nanoparticles have settled at the bottom of the beaker, the remaining solution can be poured off leaving a paste-like conductive ink. The ink is given a paint-like viscosity with the addition of a few drops of Gum Arabic. The amount of Gum Arabic added varies the viscosity of the ink until it is similar to a typical inkjet printer ink.

The Copper Sulfate ink is tested using a clean HP C6602 inkjet cartridge against the design requirements. The ink is first tested using no annealing process to bind the ink to the substrate. Investigation into heating the substrate to anneal the ink or possible chemical annealing is considered in section 3.1.2, Printing Substrate research. Refer to section A1.1 – Bibliography for sourcing information on the manufacturing of the ink.

3.1.1.4 Silver Nitrate Ink

The Silver Nitrate Ink is difficult to produce and expensive. It has a long production time requiring expensive materials and manufacturing equipment. The Silver Nitrate Ink promises to be the ink with the lowest resistivity and the smallest particulates. Considering the difficulty and cost to produce, an easier to produce ink might be selected over the Silver Nitrate Ink as long as the maximum resistivity design requirement is met.

Silver Nitrate Conductive Ink is produced by dissolving 20 g of Silver Nitrate, 1.915 g of Acrylic Acid, and 40 g of Diethanol Amine (DEA) into 50 g of water. This solution is left at room temperature for 20 hours for the Silver nanoparticles to begin to form.

After the nanoparticles have started to form, they are grown by use of a heated sonic bath at 65° Celsius for 1.5 hours. Coagulation of the nanoparticles is accomplished by adding 300 mL of Ethanol. The solution is then spun in a centrifuge at 9000 RPM for 20 minutes.

After centrifugation, the solution is poured away from the compacted particles. The compacted particles are dispersed using water and then filtered using a syringe filter.

Hydroxy Ethyl Cellulose (HEC) is a cellulose binder that is added to the filtered ink to bind the particles together. The ink is then blended using a homogenizing mixer and then allowed to evaporate until the desired viscosity is reached.

Production of the Silver Nitrate Ink is dependent on the group having the ability to use required equipment. The ink is tested using a clean HP C6602 inkjet cartridge against the design requirements. Refer to section A1.1 – Bibliography for sourcing information on the manufacturing of the ink.

3.1.1.5 Reduced Charcoal Ink

The Reduced Charcoal Ink is inexpensive to produce and is the simplest option for conductive ink. It yields a paint-like paste that will stick and dry to paper.

Ordinary charcoal briquettes are burned in a fire pit or BBQ grill until they are reduced to pieces of electrically conductive carbon. Individual pieces of charcoal are measured using a multimeter to determine the pieces that have the lowest resistance. The charcoal pieces with the lowest resistance are placed in a blender with water and blended until all large pieces have been crushed.

The charcoal particle solution is poured into a glass and left to settle for a few hours. The heavier carbon will collect on the bottom while the lighter water will float to the top. Clear water is removed from the glass using a syringe in an effort to take out as much water as possible.

A teaspoon of water based black paint is added to the concentrated carbon particle solution for a dark color. A teaspoon of Elmer's glue is added to the solution to bind the particles together and allow them to dry and stick onto paper.

The Reduced Charcoal Ink is tested using a clean HP C6602 inkjet cartridge against the design requirements. The ink, because it is comprised of blended charcoal briquettes, may pose an issue with clogging the HP C6602 inkjet print head. Refer to section A1.1 – Bibliography for sourcing information on the manufacturing of the ink.

3.1.1.6 Graphite Shaving Ink

Graphite Shaving Ink is inexpensive, but it is tedious and time consuming to produce. A razor blade is used to shave graphite particles off of a standard graphite pencil. A small amount of water and wood glue is mixed in with the graphite particles to form a paste that will stick and dry to paper.

The Graphite Shaving Ink is tested against the design requirements using a clean HP C6602 inkjet cartridge. Clogging of the print head is a risk to be considered for meeting the design requirements. Refer to section A1.1 – Bibliography for sourcing information on the manufacturing of the ink.

3.1.1.7 Graphite Glue Ink

The Graphite Glue Ink is similar in concept to the Graphite Shaving Ink. It is also inexpensive, but without the tedious and time consuming nature of the Graphite Shaving Ink.

Carbon Graphite is purchased in a fine powder form is mixed with Liquid Tape glue based on the desired consistency. According to the article cited in section A1.1 – Bibliography, a mix of 1.5 part Carbon Graphite to 1 part Liquid Tape gives the least resistivity while allowing the paste to stick and dry.

Various mixtures of the Carbon Graphite to Liquid Tape are to be tested with a clean HP C6602 inkjet cartridge. Considering that both the Graphite Shaving Ink and the Graphite Glue Ink have some form of glue in them, there is a risk of clogging the print head. Therefore, there is a risk that these ink choices do not meet the design requirements. Refer to section A1.1 – Bibliography for sourcing information on the manufacturing of the ink.

3.1.1.8 Silver Acetate ink

Based on research into conductive inks, the Silver Acetate Ink appears to be the best option. It is moderately difficult and expensive to produce, but looks to offer the highest electrical conductivity. It is produced using a chemical reaction that produces elemental silver from a particulate free liquid that does not clog the HP C6602 inkjet print head. It can be annealed to either a silicon or glass substrate by heating, thus forming a continuous trace with low resistivity that is resilient from being scratched off.

The Silver Acetate Ink is produced by combining 2.5mL of Ammonium Hydroxide with 1 gram of Silver Acetate using a magnetic stirrer. Once mixed, 0.2mL of Formic Acid is added to the mixture one drop at a time using a syringe, mixing between each drop. The solution is left to settle for 12 hours allowing the larger silver particles settle at the bottom.

Using a 0.2µm syringe filter, the larger silver particles are filtered from the solution. The final ink is tested using a clean HP C6602 inkjet cartridge. The solution leaves behind elemental silver upon annealing via heating the substrate

with the printed design to approximately 200 degrees F. Refer to section A1.1 – Bibliography for sourcing information on the manufacturing of the ink.

3.1.2 Conductive Ink Decision

Based on the research that was carried out by the group, our initial conductive ink selection is the Silver Acetate Ink. The Silver Acetate Ink provides a cost effective ink that can be produced in a reasonable amount of time. It is particulate free, which satisfies the requirements of the ink jet print head. The Silver Acetate Ink has a very low resistivity as the chemical reaction used to produce it yields elemental Silver. The resistivity of the Silver Acetate Ink can be reduced further by annealing it, which also bonds the ink together into a strong finalized state.

3.1.3 Printing Substrate Choices

Research into various processes to produce conductive ink is carried out. They are produced and tested against the design specifications to identify plausible choices. The optimal ink variety is selected from possible choices by choosing the ink with the lowest resistivity (Ω/cm).

3.1.3.1 Printing Substrate Design Requirements

- Ink that has been printed from the HP C6602 inkjet print head must be able to form a continuous line on the substrate without leaching into the material and spreading out.
- The substrate must be able to be heated to appropriate annealing temperatures when required by ink selection.
- The substrate must have a high resistivity so as to allow current to flow only through the printed ink and not through the substrate.

3.1.3.2 Printer Paper

Printer Paper is the cheapest option for printing substrate. It is only appropriate when paired with an ink that has a paint/glue viscosity as a liquid ink would seep into the paper and spread. Printer Paper is unable to be heated to annealing temperatures, thus eliminating the ink types that require heat.

While it is unlikely that printer paper can be used in the design, it will be helpful for studying the different ink types. If printer paper is successfully implemented in the design, it will make for a material that is easily procured by the user.

3.1.3.3 Photo Paper

Photo Paper is similar to Printer Paper in terms of functionality in the design, but with a few additional benefits. It is also only appropriate when paired with an ink that does not require annealing as it would present a fire hazard when heated to 200 degrees F.

Photo Paper is coated with chemicals that are designed to smooth the rough surface of standard printer paper. The chemical coating in effect, holds the ink in place, and prevents the printed ink from bleeding. If an ink is selected that does not require annealing, Photo Paper is a strong contender for ink substrate selection.

3.1.3.4 PET Transparency

PET transparency is a thermoplastic polyester film that possesses great qualities. It is also known as Mylar® Film, which has a large range of uses. For the purposes of this project we will be using it for printing. Since this polyester film is heat resistant up to 440 degrees F, it will be able to go through the process of annealing.

This film could be used with multiple kinds of inks, because of its chemical composition and ability to resist heat. In order for it to be used for printing, the Mylar® Film must go through a special coating process. This might make this option more expensive than previously mentioned substrates.

3.1.3.5 Canvas

Canvas is a material that would be interesting to implement as it is flexible and can be used to make wearable electronics. Feasibility testing into the effectiveness of canvas to retain a continuous conductor made of ink that can withstand bending forces will be carried out.

3.1.3.6 FR4 Silicon Substrate

FR-4 Silicon is a material that is constructed out of multiple plies of epoxy-resin impregnated woven glass cloth. Used mainly because it can satisfy the electrical and thermal application needs, this substrate is also used in PCB construction, but it contains a copper foil. For the inks that will be utilized, there is no need for the copper foil.

The thermal properties of this substrate make it a great option for the silver acetate ink. The silver acetate ink will need to go through the annealing process; therefore the substrate needs to withstand at least 200 degrees F.

3.1.3.7 Glass

Glass is a promising material for the design as it is a smooth surface that can withstand the annealing process. It provides a solid insulating surface for the conductive ink to adhere to without leakage current into the substrate.

3.1.3.8 Acrylic Film

Acrylic Film is also a promising material because, like glass, it provides a smooth surface that can withstand the annealing process. It is also insulating, providing for a good circuit base. Acrylic can be purchased at various thicknesses and is

cheaper than glass in certain cases. Acrylic film can also be flexible, which would allow for testing of the conductive ink's ability to flex.

3.1.4 Printing Substrate Decision

Based on the research that was carried out by the group, our initial substrate selection is glass. Glass is a good insulator that will minimize leakage current between conductive traces. It is strong and able to withstand the heating associated with the annealing process of the Silver Acetate Ink. Glass is rigid and inflexible which will reduce the risk of cracking the conductive ink traces, once printed. In addition to the electrical and material property benefits, glass is transparent, which aids in the ability to read the layout of printed circuit traces.

3.2 EXISTING PROJECTS & PRODUCTS

Within the following section, the design of a previous senior project regarding the use of Arduino development was explored, because of its similarity towards the Universal Circuit Fabricator's design plan to use Arduino hardware and software libraries. It was also used as a reference concerning how to fabricate a PCB after the microprocessor prototyping has been completed.

The section will also cover two different designs found on Kickstarter that use electrically conductive ink to print circuit traces. Since their idea resembles the UCF, their design was explored and taken into consideration towards the group's initial design. From this research, we discussed the general choices and decisions regarding their choice of substrate and conductive ink, as well as how they designed their print heads and servomotors.

3.2.1 Transit Tracking System

The previous design that was most helpful to UCF's design aspect was the Transit Tracking System that was developed for the University of Central Florida in Spring 2011. Their project was a system designed for the university's shuttling system in order to monitor and provide real-time communication from the shuttle vehicles into a host computer or web server. Figure 1 below shows the custom built PCB that was designed for use with the Transit Tracking System project.

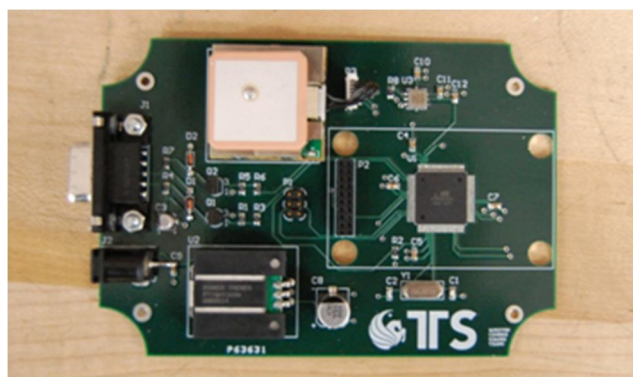


Figure 1 – Transit Tracking System Senior Design (pending permission from design team)

While their idea does not resemble the Universal Circuit Fabricator's functions, their design used the Arduino microcontroller code platform, which is one of the microcontrollers that will be used to prototype our initial design. They also fabricated their own printed circuit board from their Arduino microprocessor and breakout boards, which was used as a reference towards how the UCF's design could also be fabricated into its own PCB for ease of use.

3.2.1.1 Transit Tracking System Design

The Transit Tracking System design was comprised of two primary subsystems, which were comprised of a monitoring module and a data collection and processing system. Within the monitoring module were a Global Positioning System (GPS), an accelerometer, an RF transceiver, and finally, the microprocessor itself. While the function of their project was not similar to ours, we were interested in how they prototyped these subsystems within an Arduino microcontroller and various breakout boards, into a final product of a printed circuit board.

From their project, they decided that by using Arduino as a prototype tool first, the detailed design work would be more cost effective and require less time. They would not have to create a custom PCB for each prototype, and instead use the Arduino libraries and breakout boards to modify their design. Software development difficulty was also reduced since they could use the Arduino enabled software libraries to decrease coding difficulty and debugging time.

As stated above, the main component to their hardware development plan is the Arduino microcontroller development board, which also provided the software development libraries used for their initial design. The board included a microprocessor, I/O pins for various breakout boards, as well as a USB port for access to a computer to write software code. The I/O pins allowed Arduino breakout boards to be easily connected and tested, such as their usage of the accelerometer, GPS, and RF Modem breakout boards. Within the context of the Universal Circuit Fabricator, we plan to use different types of breakout boards within our main microcontroller in order to prototype our initial design.

Within the software development side, the microcontroller code utilized the Arduino Development Environment, which allowed access to Arduino library functions, as well as functionality for the several breakout boards that were incorporated into their design. The environment also had basic functionality in order to compile, run and flash the code into the Arduino microcontroller. Finally, in order for data processing, their code was developed in Microsoft Visual Studio C++, and was written in C++ in order to automate and organize the variables and functions they used to process and collect data from the transceiver.

Most importantly, the Transit Tracking System design made use of a fabricated printed circuit board which was used for their final product. While Arduino was used for prototyping their initial design, after verifying that their design worked, they created a circuit schematic incorporating their various subsystems as well

as the proper and necessary pin connections that they needed. Their design will be used as a reference for the Universal Circuit Fabricator in order to create the final design into its own printed circuit board to allow a compact design that will encompass all aspects of all the subsystems.

3.2.2 AgIC Printer Kickstarter Design

The motivation of the AgIC printer resembles the motivation of the Universal Circuit Fabricator. Both projects were created to bypass the messy breadboard stage of prototyping to make designing and prototyping circuits much more available to the public. Figure 2 shows a picture of a circuit design that was printed using the AgIC printer. This product is what the Universal Circuit Fabricator will attempt to mimic.

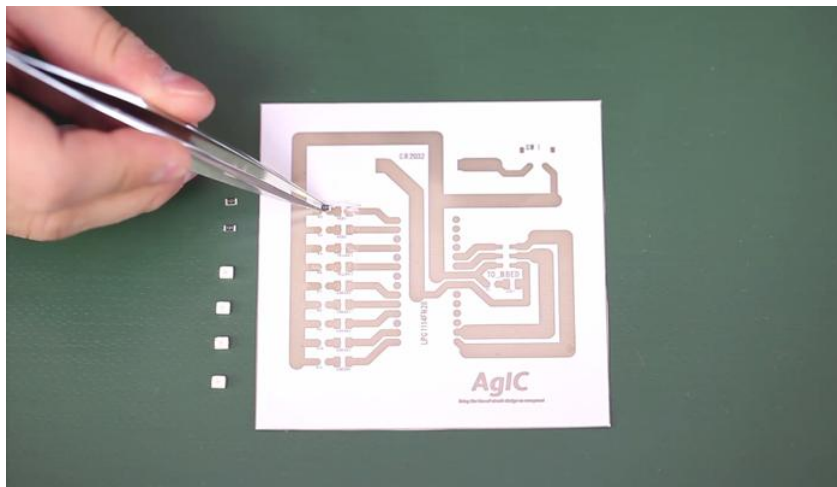


Figure 2 – AgIC Printer (pending permission from AgIC)

The AgIC printer design resembles the function of the Universal Circuit Fabricator in that it uses electrically conductive ink to bypass the breadboard in order to print circuits. However, their design wasn't its own machine, but rather a kit that transformed a typical home inkjet printer into a printer that used silver nanoparticle ink to print two dimensional circuit traces that can be printed at home.

3.2.2.1 AgIC Printer Design

In regards to their design, the AgIC printer made use of an existing inkjet printer like what could be found at home and modified it with its own custom print head using conductive ink to print conductive circuit traces. Their ink contains silver nanoparticles that allowed conductivity. Unlike our initial design, the only thing modified in their project was the print head itself.

Their reasoning towards modifying just the print head while keeping the normal inkjet printer was that home inkjet printers have high enough precision as it is in regards to printing. Therefore, by just changing the print head, they had the

ability to create circuit patterns that could be as precise as anything printed with a consumer home printer.

The ink that they decided to use for their design was made out of silver nanoparticles using a method called sintering. Sintering is the process of forming a solid mass of material by heat and/or pressure without melting it into a liquid. By using a process called chemical sintering; they used silver nanoparticles dissolved into a solvent, which after drying became electrically conducive. After being exposed to air, the conductivity continued to increase. Their research also discussed the selection of a print head, since they have to choose a nozzle that would support the small silver nanoparticles and would not be stuck on the print nozzle. Finally, their research also discussed the usage of a printing substrate, which through experimental testing they found that inkjet printer paper and glossy photo paper were effective in printing their silver ink.

From their research, the group gained valuable insight towards our decision in our own ink and substrate choices, which factored into our final decision regarding the ink and substrate that we will use for the initial and final design of the Universal Circuit Fabricator.

3.2.3 The EX Rapid 3D Circuit Printer

Like the AgIC Printer above, the function and motivation of the EX Rapid 3D Circuit Printer is similar to the Universal Circuit Fabricator. Like the UCF, it allows hobbyists and the general public to skip the breadboard prototyping. However, unlike the AgIC printer, the EX is a machine that prints 3D circuit boards by layering silver nanoparticles into a suitable substrate. Also different from the AgIC printer is that instead of modifying an existing home inkjet printer, the EX Rapid 3D is its own machine created from scratch which doesn't just modify the print head but instead is an original design. Figure 3 shows a picture of the EX Rapid 3D Circuit Printer. It is a final product that has been designed well past a functioning prototype.



Figure 3 – EX Rapid 3D Circuit Printer (pending permission from Cartesian.co)

3.2.3.1 EX Rapid 3D Circuit Printer Design

Within their design, the EX printer works by layering silver nanoparticles through a process that resembled inkjet printing into most suitable substrates. Their design included their own custom control software called ArC, which allows users to input a bitmap (.bmp) or portable networks graphic (.png) of a simple image into the software. The ArC will then use that input image to generate a control code for the EX printer that will interface with the printer for movement of the two print heads. Their software was coded using Python in order to facilitate their open-source development. Python is an open source, high level programming language that is used extensively in web programming, GUI development, software development, and system administration.

Within their actual physical design, they used a print bed made of laser cut acrylic, and their machine includes multiple servomotors that allowed movement in multiple directions. Like a typical inkjet printer, they made use of a carriage and roller assembly that included a print head stepper motor that moves the print head assembly (print head and ink cartridge) along the chosen substrate. However, unlike typical inkjet printers, their three dimensional design also had another set of servomotors that moved laterally in the y-axis direction. This allowed greater precision since the substrate did not have to move, only the print heads themselves. While their final design was not discussed in detail within the electronics side, they made use of their own hardware design using a printed circuit board specifically designed for the function of printing.

What was found most relevant towards the UCF's design was how they used user created images, which were then imported to their 3D printer. This subsystem is also part of the Universal Circuit Fabricator's design, and gave the group an idea regarding how it can be implemented for the initial prototype. The EX's use of servomotors also served as a working reference toward how the UCF's own print heads can work to print its own circuit traces. Finally, their usage of typical inkjet cartridges in order to do their printing gave the group the idea that the ink head does not need to be recreated. Instead, a typical home inkjet cartridge can be used and then the best choice of conductive ink could be inserted into empty cartridges in order to print the electrically conductive circuit traces. This allowed the group to focus on the servomotor and data processing subsystems, instead of the design of a completely new print head.

4 Project Architecture Selection

The architecture of the prototype design must be selected carefully to ensure that the required hardware and software can be integrated into a functioning system.

Hardware Architecture selection involves selecting an appropriate prototyping microcontroller board so that subsystems can be integrated and tested individually for functionality before the final PCB is ordered.

The Software Architecture selection involves selecting an appropriate Integrated Development Environment (IDE) that can successfully program the microprocessor with the UCF's source code operating software.

The System Architecture will evolve as the project is developed. Initially, the subsystems will be integrated to a prototyping microcontroller board using separate hardware breakout boards. Breakout boards are built and sold commercially for use by hobbyists to control input and output systems using a prototyping microcontroller board.

Once the functionality of the Universal Circuit Fabricator has been verified using the prototyping microcontroller board and breakout boards, the system architecture will be redeveloped to eliminate the prototyping and breakout boards by designing and ordering a custom PCB. The PCB will contain the selected microprocessor as well as all circuitry that was originally contained on the separate breakout boards. Creating a custom PCB that integrates all of the breakout boards and the microprocessor onto a single chip will reduce space needed in the enclosure of the final product. In addition to reducing space, a custom PCB will reduce the complexity of connecting several breakout boards. Replacing external connections between multiple boards with embedded connections on a single board will reduce the likelihood for continuity issues and ground faults, thus reducing debugging efforts.

4.1 POSSIBLE ARCHITECTURES CHOICES

This section will cover the possible hardware and software architecture choices that was researched and considered for the Universal Circuit Fabricator. All relevant information was collected and was then used and utilized to go over possible design configurations before the primary decision for the initial design was reached. For the final design, tweaking and changing the prototype will still take place before committing to a finalized printed circuit board to allow compact design with access to all of the UCF's subsystems.

4.1.1 Possible Hardware Architecture Choices

Within the design of the Universal Circuit Fabricator, through research, we have established that the use of a microcontroller would be a necessity within our initial prototype design before being manufactured into a printed circuit board. This is because of the fact that a microcontroller is designed to execute a stored

set of instructions defined by the programmer, which is suitable for creating a printer. Also, it has the ability to read and write data from memory, which will be important for the data processing and input output system that the UCF has.

However, our choice of microcontroller was debated within our design. The advantages and disadvantages between the Arduino and the MSP430 microcontroller code platform are discussed below along with the final decision for the prototype decision. After prototyping, regardless of the microcontroller used, the final design will be fabricated into a PCB.

4.1.1.1 Arduino

The Arduino single board microcontroller was first discussed as a possible choice for prototyping the Universal Circuit Fabricator. Used heavily by hobbyists, it is an open-source hardware built around generally an 8-bit microprocessor or 32-bit ARM processor. With regards to the UCF's prototype design, the Arduino UNO 328P microcontroller board was chosen as a possible prototyping tool.

The UNO 328P has 16 analog inputs, 1 UART, 32 digital I/O pins, a 16 MHz crystal oscillator and can be powered both by a power jack and a USB connection. These specifications allow it to be a general multipurpose device that has enough customizability to be used for the UCF initial design. Using the extensive Arduino software libraries, the UNO 328P could be connected to a computer through USB and be programmed using C and C++ languages. Because of its multiple I/O connections, the Arduino hardware development also supports using shields-breakout circuit expansion boards that could plug into the main board for added features. Within the context of our initial design, these breakout boards could include the print head, motor control, an LCD screen, and data receiving and processing to transmit user drawn traces from a picture to the UCF itself. Table 1 lists the specifications of the Arduino UNO single board microcontroller.

Table 1 – Arduino UNO Specifications

Arduino UNO	
Microprocessor	ATMega328P
Operating Voltage	7-12V
Digital I/O Pins	14
Analog Input Pins	6
Flash Memory	32KB
SRAM	2KB
EEPROM	1KB
Clock Speed	16 MHz

4.1.1.2 MSP430

The Texas Instruments Launchpad MSP430 was next researched as a possible prototyping option for the Universal Circuit Fabricator. Like the Arduino UNO, the MSP430 is also a single board microcontroller built instead around a 16-bit processor. Also used by hobbyists, it is an open-source hardware platform with an included software development library called Energia, which is comparable to Arduino's own software development platform. In regards to the UCF design, we have chosen the MSP-EXP430F5529LP as it is comparable to the Arduino UNO 328 microcontroller board

Like the Mega 328, this MSP430 model has 40 digital I/O Pins, 2 UARTs, and 16 analog inputs. These specifications allow it to be useful in many general operations as it can also be powered by USB and by a typical power jack. Also comparable to the Arduino is the Texas Instruments Energia software, which is featured as on-board emulation on the microcontroller, which means code can be easily be programmed and debugged directly on the board without any additional tools. Finally, the Launchpad is compatible with MSP430 breakout boards, which can plug into the main microcontroller for added features like the print head, motor control, LCD screen, and our data input and processing system. Table 2 lists the specifications of the MSP-EXP430F5529LP Launchpad experimenter board.

Table 2 – MSP430F5529LP Specifications

MSP430 Launchpad	
Input pins	20
Clock Speed	25 MHz
Memory	128 KB Flash
Onboard Memory(RAM)	8 kB
A-D Converter	12-bit

4.1.2 Possible Software Architecture Choices

In the sections below, the group explored and researched all possible software architecture options for developing the Universal Circuit Fabricator. Three software choices were explored, with the Arduino IDE, the Texas Instruments Code Composer Studio and the Energia open software IDE for the TI MSP430.

4.1.2.1 Arduino IDE

Arduino Integrated Development Environment (IDE) is an open-source software development environment. The environment is written in Java and based on Processing, avr-gcc, and other open source software. It is compatible with all of the Arduino versions available. The programs written in this environment are

written in C or C++. The Arduino IDE comes with some software Libraries, a main one is the library called “Wiring”. This library makes it easier to use input/output operations. This makes it an ideal software environment for people that do not have an extensive background on software programming.

The use of this environment will be used with the Arduino Uno Dev-Board, to program the InkShield breakout board. Once, the program is deemed suitable for prototyping purposes it will be transferred into another software environment.

4.1.2.2 Texas Instruments Code Composer Studio

Code Composer Studio is an Integrated Development Environment (IDE) for Texas Instruments embedded processors, including their microcontrollers. Based on the Eclipse IDE, it allows easy development and debugging of all Texas Instruments embedded applications, including the MSP430. A bi-lingual platform, it supports multiple languages in C, and C++. It also includes a source code editor, project build environment, and a debugger.

Code Composer Studio has been in use for many years, and as a result, software libraries and tutorials have created a feature-rich development for all embedded applications. In regards to the Universal Circuit Fabricator’s design, this provides an ease of use in developing our Graphical User Interface with our MSP430 initial design. Once the program has been up and running within our prototyping purposes, the code will be transferred into a PCB for compact usage.

4.1.2.3 Energia

Energia is an open source electronics platform that is a comparison to the Arduino Integrated Development Environment. However, instead of being based on the Arduino hardware platform, it uses the Texas Instruments MSP430 based Launchpad. Like the Arduino IDE, the programs written in this environment are written in C or C++. Also like the Arduino IDE, Energia makes extensive use of software libraries which makes it easy to create input/output operations. This creates an ideal software environment for casual hobbyists and for the design for the Universal Circuit Fabricator.

The use of this environment will be used with the MSP430 Launchpad prototyping board to program the initial design of the Universal Circuit Fabricator. Once the program is working for the initial design, the code can be flashed to a printed circuit board for compact usage.

4.2 SELECTION OF INITIAL DESIGN ARCHITECTURE

The Initial Design Architecture Selection section discusses the thought process towards selecting the initial design of the Universal Circuit Fabricator. Below, the hardware architecture and the software architecture selection will be selected and debated.

4.2.1 Hardware Architecture Selection

In the initial design of the Universal Circuit Fabricator, the Arduino UNO was selected as the microcontroller hardware choice. This is due to the fact that comparing to the MSP430, the Arduino microcontroller is more easily accessible for purchase, as well as having the same ease of use. In regards to hardware, there are microcontroller breakout boards that also have the same type of functionality that the UCF needs for its design. For each subsystem in the UCF, there exists an Arduino breakout shield that can perform the needed function.

Essentially, there is no disadvantage in using the Arduino UNO over the TI MSP430 Launchpad. The Arduino has a lower price point, is more accessible, and can be used exactly as though it was a very easy project with the new software libraries. With these reasons, the Arduino UNO was chosen as the Universal Circuit Fabricator's main prototyping microcontroller.

4.2.2 Software Architecture Selection

For the initial and final design of the Universal Circuit Fabricator, since the Arduino UNO was selected as the microcontroller hardware design choice, within the software design, the Arduino Integrated Development Environment was subsequently chosen for the hardware. As stated above, the software development environment function is specifically for Arduino hardware, and makes extensive use of software libraries that make it easy to create functions suited for design purposes.

The group also has prior experience in regards to the use of Arduino hardware within their classes and hobbies. With previous experience and due to the fact that the Arduino IDE has been in use for many years, there are multitude resources and tutorials available online and various books in order to provide an ease of use for the initial design.

After prototyping, like the hardware, the software code will be flashed into a designed printed circuit board with an Atmel ATmega328p microprocessor in order to provide compact usage and allow all subsystems of the Universal Circuit Fabricator to exist in one board for further manufacturing and assembly.

4.3 OVERALL SYSTEM ARCHITECTURE

The overall system architecture will rely on a mix of hardware and software to achieve its goals. As stated above, from the hardware architecture decisions, the Arduino UNO was chosen for initial prototyping due to its ease of use and familiarity within the whole group. In regards to software, Arduino family also includes easy access to the Arduino IDE as well as various amounts of open source libraries that supports a large amount of functions for use within the Universal Circuit Fabricator systems.

For the UCF's initial prototype, breakout boards will be used for the individual subsystems within the UCF, such as the Inkshield breakout board, the Capacitive

Touch Screen breakout board, or the motor control system breakout board. Having individual and separate breakout boards will be useful for prototyping, as they would make troubleshooting the initial design very easy.

In regards to software development, with the choice of the Arduino IDE, the development environment has vast software libraries to help write code for both the initial and final design of the UCF. Arduino is open source, and is comparable to the C language, which the group is very familiar with. The Inkshield also uses code native to the Arduino IDE, which makes posting functionality very simple

After prototyping on the Arduino UNO, the group will use the Atmel ATmega328P microprocessor on the original Arduino UNO and transfer it into a self-built printed circuit board. Having the design on a printed circuit board provides compact access, as well as ease of use, as there would be no need for separate breakout boards unlike the initial prototype. The software code that was used for the initial design can also be easily flashed to the final PCB, as the same processor will be used for both the initial and final design.

5 Prototype Hardware Design

The following sections go over how the initial prototyped design was established and researched, and what parts were acquired in order to prototype the Universal Circuit Fabricator. First, the main overall system level design of the UCF will be explained, before the next section, where individual component details and configurations of each subsystem within the UCF will be explored. Block diagrams will also be used to provide an easy to understand figure to convey the input/output relationships between each subsystem. The final designed printed circuit board will be discussed in Section 8

5.1 SYSTEM LEVEL DESIGN

The following sections go over how the initial prototyped design was established and researched, and what parts were acquired in order to prototype the Universal Circuit Fabricator. First, the main overall system level design of the UCF will be explained, before the next section, where individual component details and configurations of each subsystem within the UCF will be explored. From this section, the initial hardware design of the UCF will be gone into detail.

After prototyping, the design will be translated and shifted into a printed circuit board. The printed circuit board will have all of the sub systems within one PCB allowing for compact usage. The data processing system, SD card input, motor control system, and the inkjet cartridge system will all be soldered into the final PCB so that all aspects of the design will be included. A custom power supply will also be included in the board to provide power for all the components. For the annealing system, a separate heated print bed will be powered separately as its wattage is higher than most of the electronic components. Afterwards, the entire PCB, motors and print bed will be placed in an enclosure along with the substrate for printing purposes.

5.1.1 Component Details & Configuration

In the figure below, the high-level diagram of the hardware configuration is illustrated, which includes the data input, microcontroller, motor control, Inkjet Cartridge control, LCD Graphical Interface, and the annealing subsystem. This diagram shows the flow of information throughout the system, in a basic level.

Initially the user will create a drawing in a circuit design program and output the file as a jpeg or a schematic diagram. From there the user will then put the file into an SD card and place it on the SD card breakout board. From there the microcontroller will read the SD card and display its contents on the LCD GUI interface.

After the user decides what they would like to print, the GUI will give the user a preview of the diagram that they would like to print. Then, they have the option to print or return to the previous menu. When the print command is accepted, the inkjet cartridge control and the motor control will take over.

When the circuit is completely finished, the annealing process will take over. This will harden the ink making it less resistive. Further details of each of these systems will be thoroughly covered in the upcoming sections.

5.1.2 Block Diagram

The following figure shows the overall system architecture and block diagram and how each of the subsystems interconnects and interacts with one another.

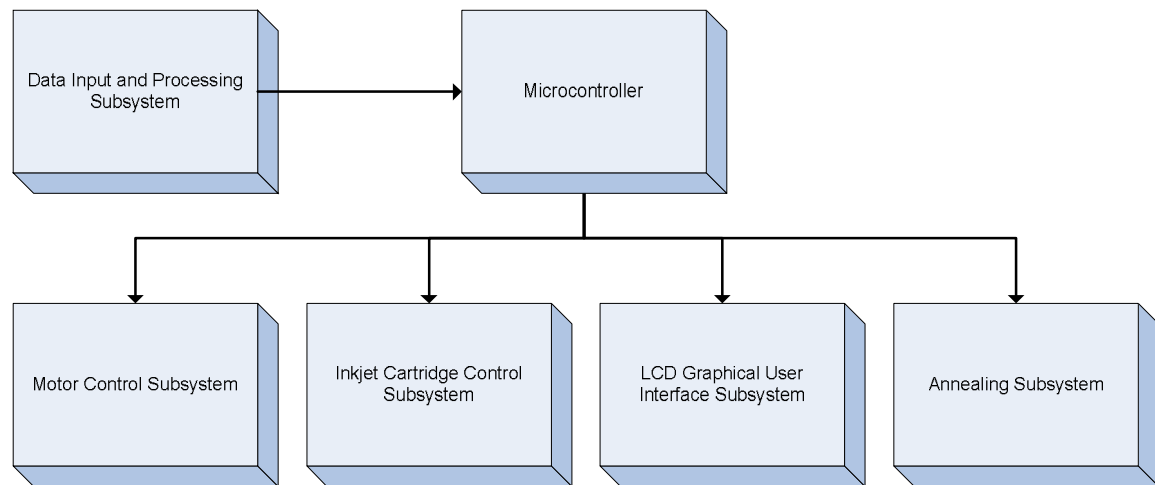


Figure 4 – System Level Design Block Diagram

5.2 INKJET CARTRIDGE CONTROL SYSTEM

The Ink Jet Cartridge Control System is responsible for controlling the flow of conductive ink from the Ink Jet print cartridge. It will rely on data fed from the microprocessor as an input to toggle the ink jet nozzles to turn on and off.

5.2.1 Component Details & Configuration

The inkjet cartridge control system is driven by the InkShield, designed by Nicholas C. Lewis. The InkShield essentially turns an Arduino/MSP430 into a 96 dpi print platform. The InkShield works with five output pins, user selected, from either the Arduino or the MSP430.

The Inkshield requires a 12 volt input and is powered through the output of the Dev-boards Vin pin. On the InkShield there is a power booster to increase the voltage supplied to the inkjet cartridge, to 20 volts operational value. The InkShield operates an HP C6602 inkjet cartridge that has been tested and researched for compatibility by Lewis.

The HP C6602 contains 12 nozzles, which would mean that it requires 12 pins from the Dev-board, which is not very effective. In order to overcome this obstacle the InkShield has a 4 to 16 multiplexer, to reduce the pins used to only four instead of 12. There was also the option of using a shift register which would

allow all of the nozzles to be fired at once, but the specifications sheet stated that it would not be recommended. Therefore the shift register was not a viable option, so the multiplexer was selected as the best option.

Lewis found that in order to operate the HP C6602, it requires a five micro second at 20 volt pulse in order for the nozzles to spray. It also requires an 800 micro second cool down period in order not to burn the nozzles. There is also a pulse width modulator within the circuit that can be controlled through the InkShield software.

5.2.2 Block Diagram

The following figure shows the Inkjet Cartridge Control architecture and block diagram and how each of the components interfaces with each other.

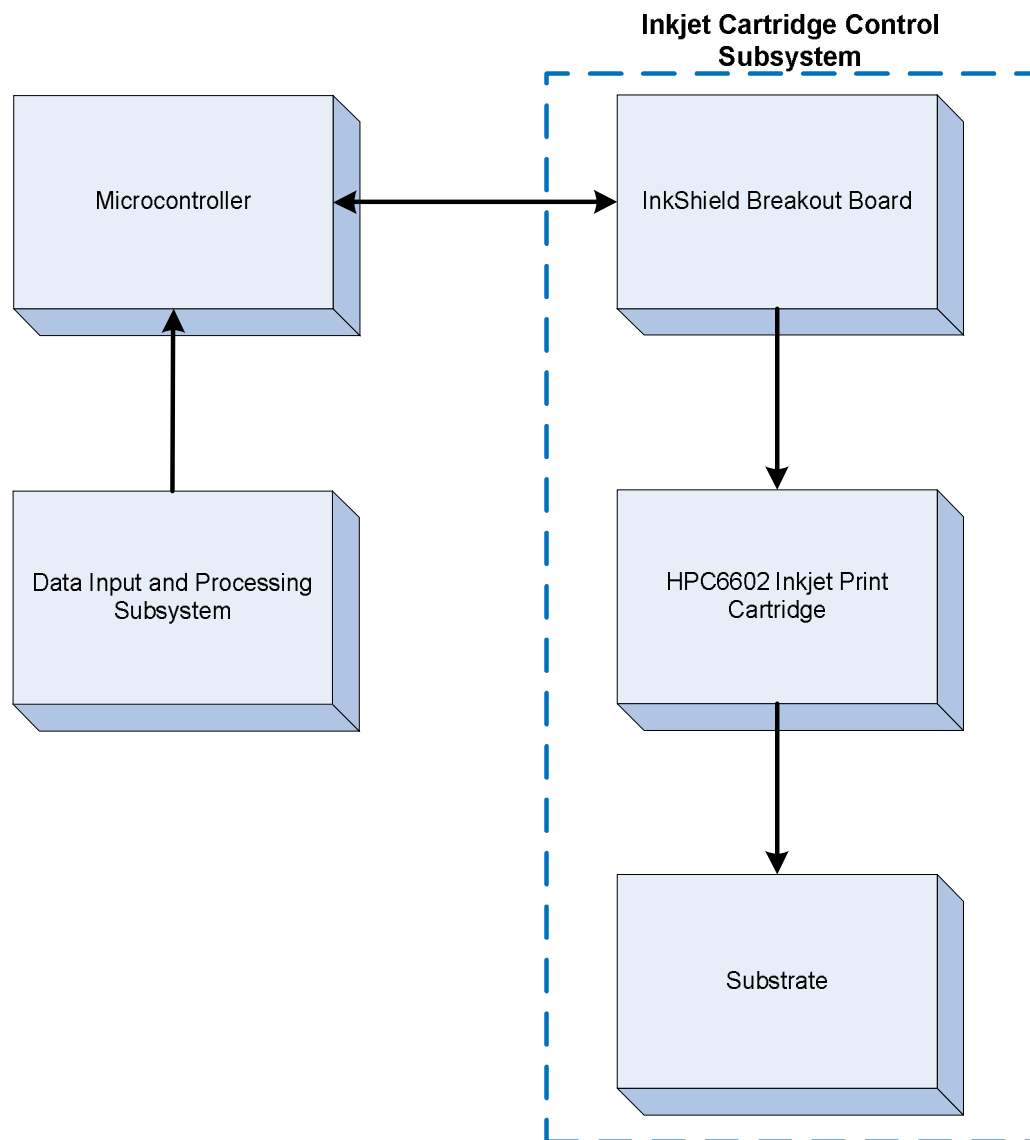


Figure 5 – Ink Jet Cartridge Control System Block Diagram

5.2.3 InkShield Parts List

This subsystem consists of three main sections: the InkShield Breakout board, the HP C6602 Inkjet Print Cartridge, and the substrate. Beginning with the InkShield, the microprocessor sends multiple signals to the InkShield. Within the InkShield the signal received will select a nozzle on the HP C6602 cartridge and it will spray ink onto the substrate.

The following table includes the subsystem and the components, which is included within the Universal Circuit Fabricator's Initial Design:

Table 3 – InkShield Board Part List

Subsystem	Components
InkShield	Transistor Darlington Array, Multiplexer Switch IC's 16-Ch, Schottky Diodes Rectifiers Vr/40V Io/1A BULK, Fixed inductors Voltage Regulators Capacitors Header Connections Operational Amplifier
HP C6602 Cartridge	Ribbon connector Cartridge holder C6602 Cartridge
Substrate	FR-4 silicon substrate Glass Acrylic Sheet

5.3 MOTOR CONTROL SYSTEM

The Motor Control System is responsible for controlling the movement of the X and Y axis stepper motors that are mounted so as to translate the print head across the print bed surface. It will rely on data fed from the microprocessor as an input to toggle the direction of the motor's movements and the amount of steps according to the circuit trace design

5.3.1 Component Detail & Configuration

The Motor Control System's main component is the Stepper Motor breakout board that is connected to the initial prototyping design. Within our final design, the breakout board will be replaced by its own subsystem within the PCB.

In our initial design, the breakout board is connected to the Arduino UNO Microcontroller. Instructions from the user will be inputted into the microcontroller, and the microcontroller will transmit that data into the Stepper Motor breakout board to create movement with the two stepper motors.

The breakout board will simultaneously move two motors, where one motor will be solely responsible for the x-axis movement, and the other will only move in the y-direction. This will ensure accuracy of the print on the substrate. After the microcontroller has transmitted that the image is done being printed, the motors will stop moving after receiving stop instructions from the hardware and reset back to its original position.

5.3.2 Block Diagram

The following figure shows the Motor Control architecture and block diagram and how each of the components interfaces with each other.

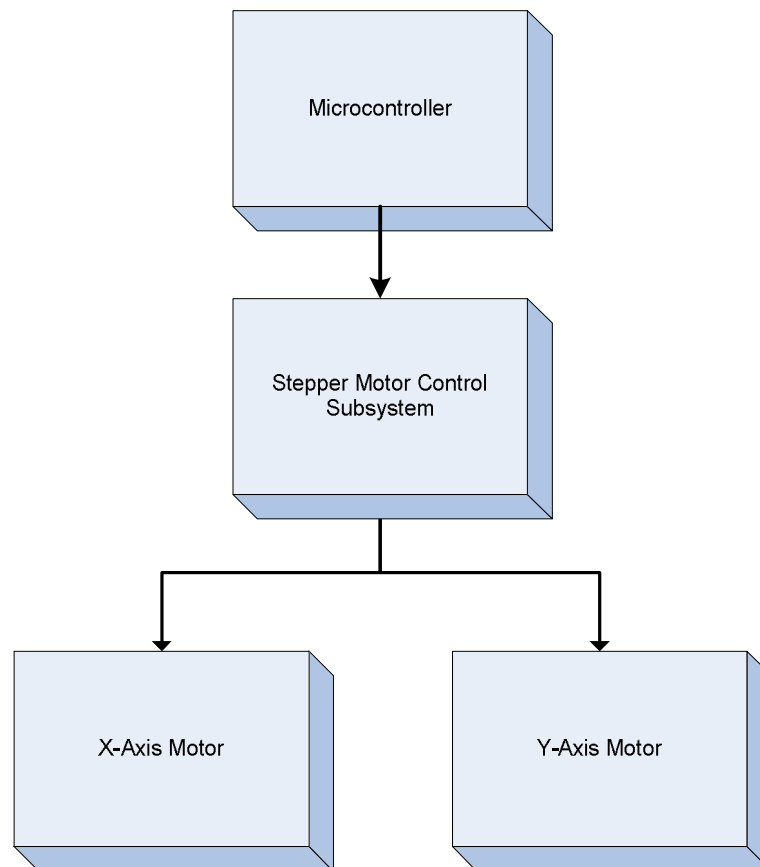


Figure 6 – Motor Control System Block Diagram

5.4 ANNEALING SYSTEM

The Annealing System is responsible for controlling temperature of the heated print bed when heating the substrate to anneal the conductive ink. It will rely on data fed from the feedback control thermocouple to the microprocessor as feedback loop to maintain a constant annealing temperature for a set duration.

5.4.1 Component Detail & Configuration

The Annealing System functions to heat the print bed after the printing process has completed. Heating of the substrate allows the Silver Acetate Ink to melt and fuse together into a continuous conductive line, which reduces resistivity. The annealing process is started when the user selects inputs the start command via the Graphical User Interface. The print head is moved away from the print bed to prevent damage to the HP C6602 inkjet cartridge. The microcontroller turns on the hot plate and begins monitoring the temperature of the substrate via a thermocouple. The microcontroller regulates the temperature of the substrate to a level defined to be optimal for annealing the Silver Acetate ink (~200 degrees F) for a defined time period. After the annealing process has completed, the hot plate is turned off and allowed to cool.

5.4.2 Block Diagram

The following figure shows the Annealing System architecture and block diagram and how each of the components interfaces with each other.

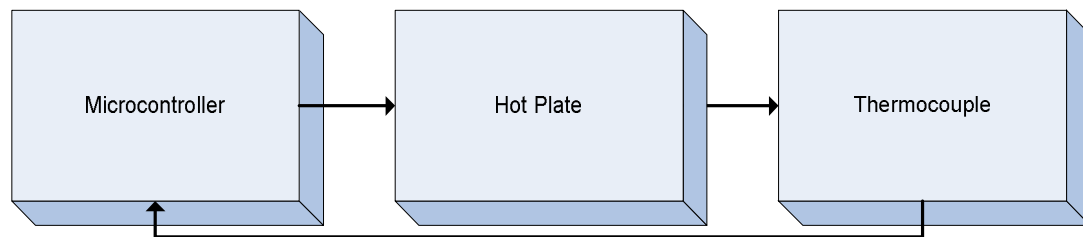


Figure 7 – Annealing System Block Diagram

5.5 DATA INPUT & PROCESSING SYSTEM

The Data Input and Processing System is responsible for controlling the microcontroller based on input from the user. It comprises of an SD Card breakout as well as a Capacitive Touch Sensor that allows the user to navigate the graphical user interface. It will rely on data from the user and must successfully process and store the input data in the software of the microprocessor.

5.5.1 Component Detail & Configuration

In regards to the Data Input and Processing System in the Universal Circuit Fabricator, the block diagram seen in the next section represents the data flow within our initial prototyped design. The block diagram has two inputs, which represents how users can either use the capacitive touch screen to create a basic circuit design trace, or input a bitmap or .jpeg file into the design.

First, a user can create a simple circuit trace image in their chosen software within their personal computer and save it as a bitmap (.bmp) or a jpeg (.jpg). This image will be saved into a Secure Digital (SD) memory card of their choice for storage.

Next, the SD card will be read by the breakout board hardware within the initial microcontroller design. The data will be processed in the SD Card breakout board for further use within the microcontroller and the Atmel ATmega328P microprocessor.

As the data is transferred to the MCU, it is processed into a series of instructions and signals that represent the design of the initial drawing, as well as data that can preview what the final printing will look like. This data will be converted visually and transferred into the graphical user interface on the LCD screen. This will allow the user to get feedback towards what their final printing output will look like.

In regards to the second input line, the user can also use the Universal Circuit Fabricator's touch screen to input their circuit design. From their user input, the transmitted data will flow into the Capacitive Touch Sensor breakout board within the microcontroller initial design. The data will get translated into instructions that will be received by the microcontroller.

Within the microcontroller, the data from the breakout board will be used as data that will go into the graphical user interface, which, like the SD card path, will show a print preview before the user decides to print. In this path, the graphical user interface will continue to act as a feedback to the user input, so that they are able to correctly see if their input matches the drawing that the LCD screen is displaying.

5.5.2 Block Diagram

The following figure shows the Data Input and Processing System architecture and block diagram and how each of the components interfaces with each other.

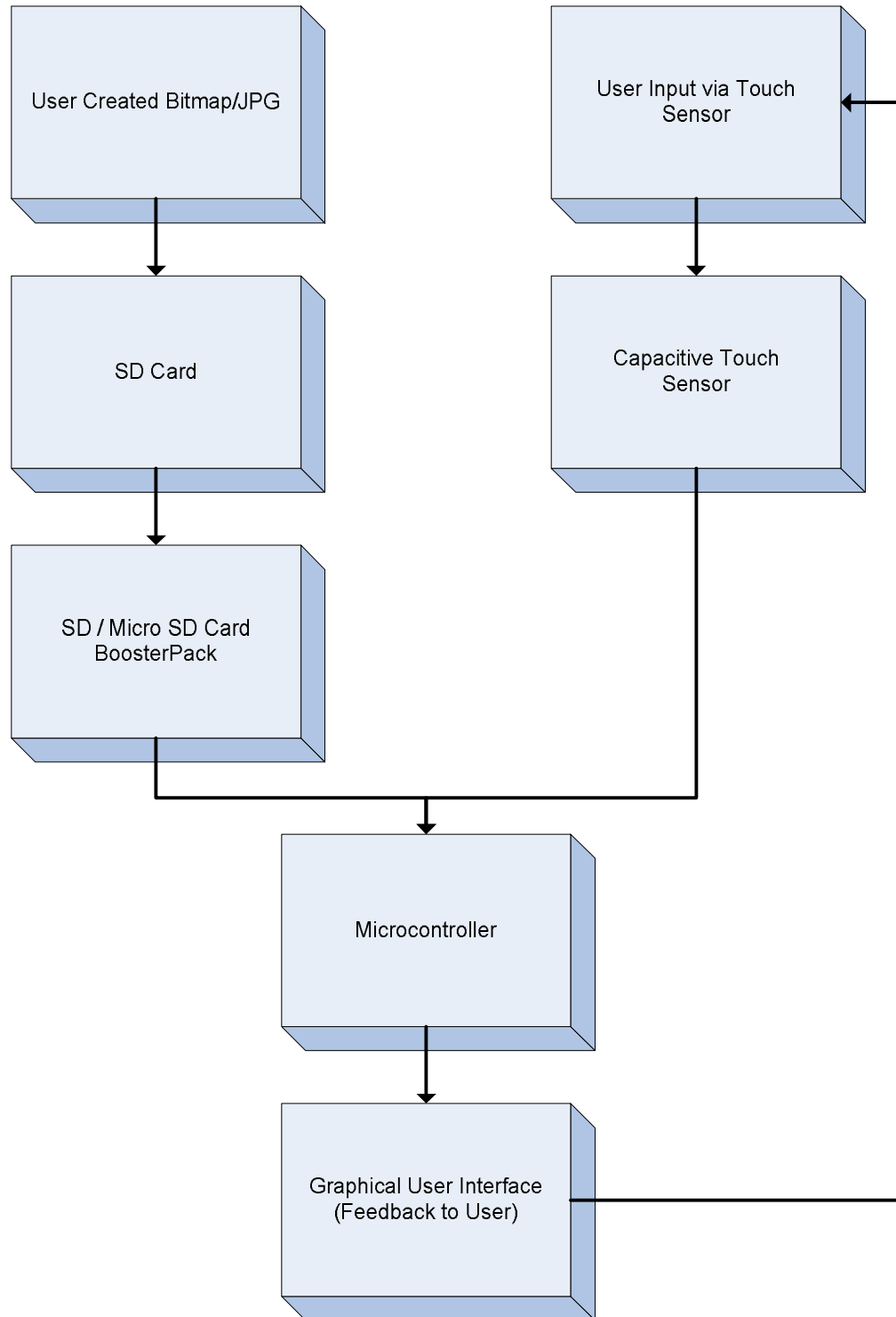


Figure 8 – Data Input and Processing System Block Diagram

5.6 MICROCONTROLLER SYSTEM

The Microcontroller System is responsible for storing the source code of the UCF's software and integrating all other systems together. It will be prototyped using a prototyping microcontroller board and once functioning, will be implementing in a PCB.

5.6.1 Component Details & Configuration

The microcontroller interface we are using in our initial design for the Universal Circuit Fabricator is the Arduino UNO with its Atmel AtMega328P Microprocessor. The block diagram seen below starts with the user input, which can either be through the SD Card breakout board or the Capacitive Touch breakout board attached to the microcontroller.

The SD Card breakout board can recognize user created basic circuit traces in either a JPEG or a bitmap format created from their own computer. After storing it within a Secure Digital (SD) card, the memory card can be inserted into the breakout board and the data to be transferred into the Atmel ATMega328P. The processor within the microcontroller will then process that data for further use.

Within the other input, the user can create a basic circuit trace with the Capacitive Touch breakout board which can be reflected by the attached LCD screen as a feedback for the user. This means that while the user is drawing their circuit trace, they can see a print preview towards what the circuit trace will look like before actually printing on the substrate. Like the first path, after being created, the data of the created trace will flow into the ATMega328P Microprocessor and will be used by the rest of the microcontroller subsystems.

Within the output bus in the next step are all the different subsystems that will use the data that is now present within the microprocessor:

First, the Inkshield Breakout Board will take the data flow from the microprocessor and start to turn on the HP C6602 Ink Cartridge in order to print on the substrate. The microprocessor will have instructions in the code for the InkShield board to start and stop spraying the ink nozzle within the ink cartridge that holds the electrically conductive ink. These instructions will stem from whether the microprocessor will read a null value from each pixel image, representing no drawing, or return a value, representing that something has been drawn in that image. If the microprocessor reads a returned value, the ink nozzle will spray and vice versa.

Within the Stepper Motor breakout board, the microprocessor will relay instructions to the hardware that will determine the start and stop of the two motors. One motor is solely responsible for the x-axis direction while the other is only responsible for the y-axis motion. The instructions will come directly from the visual data that the primary inputs from the SD Card or the user inputted touch screen circuit trace. Like the Inkshield above, the microprocessor code will

comprehend whether the bitmap will either have a null value or return an accepted value. This represents whether or not to move the motors. After the whole image has been processed, the motors will move back to their original state.

In the Temperature Control System, the microcontroller will relay information to the annealing system via a thermocouple that will recognize whether the temperature is at an acceptable state for the substrate. The microprocessor will relay a signal for the hot plate to turn on or off, and in a feedback system, after the thermocouple hits a temperature in the substrate that has successfully fused the ink into a continuous conductive trace, the signal will be sent back to the microprocessor to turn the hot plate off.

Finally, in the LCD Screen subsystem, the microprocessor will send a preview of the final print of the circuit trace before being printed to the substrate. This will feedback to the user, as it will allow them to see the print before the motors and the Inkshield system outputs to the substrate. After the user is satisfied, they will verify the accuracy of the trace and then the system will begin to print.

5.6.2 Block Diagram

The following block diagram represents the microcontroller system as all blocks in the green background. It visualizes what systems will be implemented in the PCB and how it will integrate with external hardware.

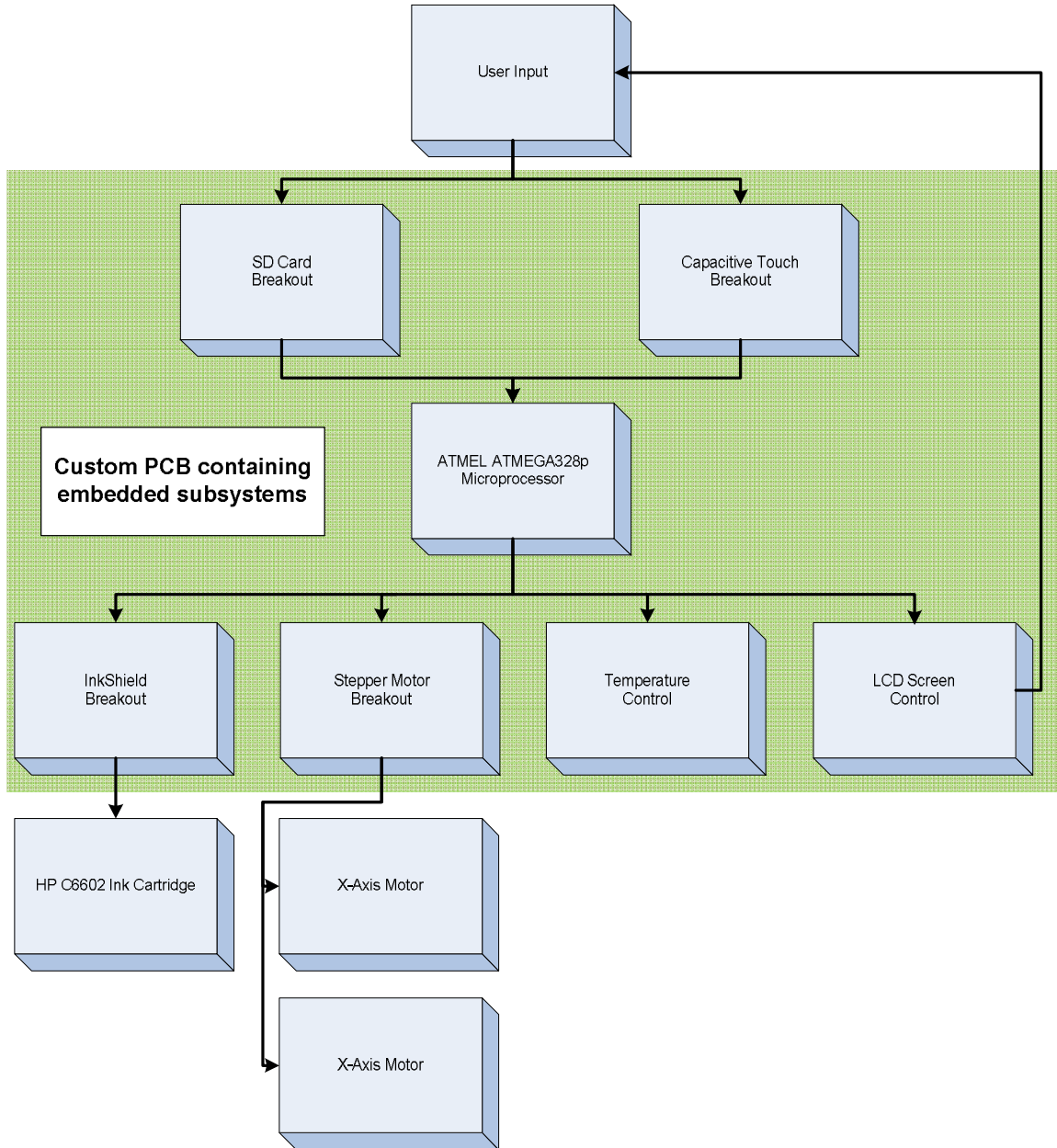


Figure 9 – Microcontroller System Block Diagram

6 Software Design

The following sections will describe in detail how the software functions within the Universal Circuit Fabricator. Since the UCF is based on a number of different subsystems, each subsystem will be gone into detail over how the software works in order to make the final printed circuit trace.

First the overall system level design will be explored, then each following subsystem, making sure to explore the coding plan, and the flow of data from the user, into the microprocessor, and finally into the different subsystems. Functions, variables, and pin assignments will also be described.

6.1 SYSTEM LEVEL DESIGN

The Universal Circuit Fabricator software functions based on multiple various functionalities. This is due to the fact that multiple subsystems are necessary in order for the UCF to function in its entirety. First, the software will have the data input and processing subsystem in order for the user to input circuit traces for the UCF to print on the substrate. The software within that subsystem will require heavy use of the processor and the buffer in order to process the data that the microcontroller will use during the printing process. The data also needs to be able to be written into the memory of the processor if the user uses the capacitive touch screen to draw their own circuit trace. Within the Energia software libraries the UCF will be running on, the software will allow manipulation and storage of the raw data that is collected.

Next, the data will enter the microcontroller itself, which in our case will use the ATmega328P processor which will be in its own PCB in the final design. Within the microcontroller code will be basic functions for functionality of the printer, such as controlling the print speed and the travel speed of the motors. Within the micro-controller code, it will also declare all of the pin numbers within the UCF subsystems for both inputs and outputs. This can be used to check if everything is wired correctly and working properly.

Finally, after the ATmega328P processes the data, the software will incorporate the various subsystems that depend on the processor, such as the motor control system, the annealing system, the LCD system, and finally the Graphical User Interface within the LCD screen. The software functions and coding philosophy of each the system level design will be described in detail below.

6.1.1 Main Function Description

Within the Universal Circuit Fabricator, one main function named `print_start` is the function that governs the entire process of printing. This function will use the subroutines that will be present in the subsystems further described in the following sections, such as functions from the motor control system, inkjet cartridge system, and the data processing system. The `print_start` function will

govern the main action of printing, and use all subsystems to coordinate the printing of a continuous circuit trace on the finished substrate.

The function is listed, along with its input and output variables, as well as a brief description of what it does in the table shown below.

Table 4 – Main Function Description for start_print

Function Name	Inputs	Outputs	Description
Start_print	(void)	(void)	Main function that resets the coordinates stepper motors and printer head nozzles before starting to print. Uses multiple functions described in the subsystems such as start_nozzle, line_pos, buffer_update, etc.

6.2 MICRO-CONTROLLER CODE

The microcontroller embedded within the Universal Circuit Fabricator will be the control center of each subsystem within our final design. The code written for the micro controller will be comprised of several different tasks. First, it will declare all the input and output pins of the several different subsystems. This will ensure that all of our input and output hardware works correctly. The microcontroller code will also contain basic configurations for the printer itself, such as print speed and travel speed. Finally, within the main code will be the different variables and functions that we will be using within the different subsystems.

The Texas Instruments Code Composer Studio and Energia were chosen for this project. The set of development tools will improve efficiency during prototyping because of its ease of use and its similarities to the C language syntax. It also includes extensive libraries and a large user base for troubleshooting.

The logical block diagram for the micro-controller software is shown in the figure below.

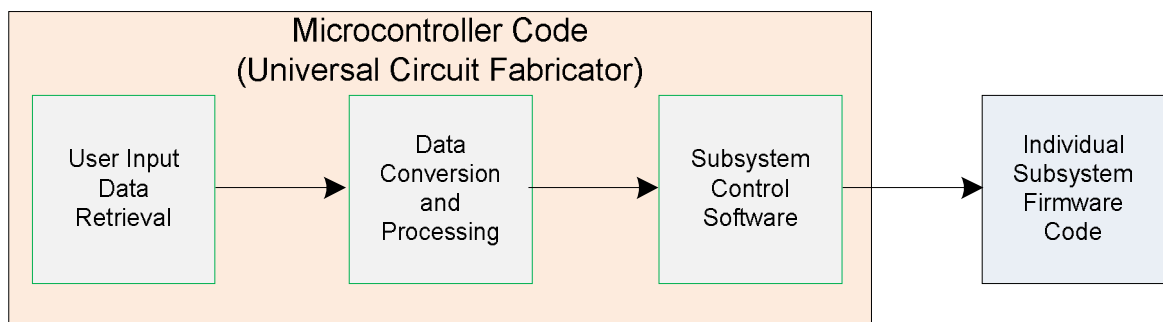


Figure 10 – Microcontroller Code Block Diagram

6.2.1 Variable and Pin Descriptions

This section describes the main variables and pin assignments present within the microcontroller code in order to perform its main functions. These variables will be used within the different subsystem codes. The Table below lists the variables and a brief description of their use.

Table 5 – Variable and Pin Descriptions for the Microcontroller

Variable Name	Type	Description
Print_DPI	int	The dots per inch value of the printer cartridge. Within the HP C6602, the default value is 96 DPI.
Print_Speed	int	The default speed the printer will print in normal operation. Higher speed will decrease the time spent but might also decrease accuracy
InputData	string	Raw data received from user input either from the SD card or the capacitive touch screen.
PackagedData	string	After processing and conversion, the packaged data will leave the microcontroller and be used for the subsequent individual subsystem codes.
X_motor_step	Int	The pin number of the microprocessor in relation to the X-axis step motor. Used as either an input or output to read/write to the motor.
Y_motor_step	Int	The pin number of the microprocessor in relation to the Y-axis step motor. Used as either an input or output to read/write to the motor.
X_motor_enable	int	The pin number of the microprocessor in relation to the x-axis step motor. Determines whether the x-axis motor is enabled or disabled.
Y_motor_enable	int	The pin number of the microprocessor in relation to the y-axis step motor. Determines whether the y-axis motor is enabled or disabled.
X_axis_dir	int	The pin number of the microprocessor in relation to the x-axis step motor. Used as an input to determine the direction of movement in the x axis direction.
Y_axis_dir	int	The pin number of the microprocessor in relation to the y-axis motor. Used as an input to determine the direction of movement in the y axis direction.

Nozzle_enable	int	The pin number of the microprocessor in relation to the inkjet print nozzle. Used as an output to turn on the nozzle on the print head to start spraying conductive ink
Heated_bed_enable	Int	The pin number of the microprocessor in relation to the annealing system bed. Used as an output to turn on the heat of the print bed for the final substrate circuit trace
SD_chipselect	Int	Relates the pin number of the microprocessor to the SD card reader on the circuit board. Used as an input in order to read a circuit trace from the SD card to be processed in the microprocessor for printing.
LCD_chipselect	Int	Relates 4 pin numbers of the microprocessor to the LCD subsystem on the circuit board. The LCD screen is used as an input in order to draw a circuit trace and also as an output for the user to see their circuit trace before printing.

6.3 INKJET CARTRIDGE CONTROL SYSTEM

Within the following section, the software design for the inkjet cartridge system will be explored in detail.

6.3.1 Software Configuration

After the user input data has been converted and processed by the microprocessor, one control line will flow into the inkjet cartridge control subsystem. This subsystem will be in charge of the print head nozzle, as well as incorporating the Inkshield Breakout Board that we used for prototyping. The nozzle will either turn on or turn off depending on the signal that is received from the microprocessor. The print head will also work tandem with the stepper motor system. When receiving a signal to move, the print head and inkjet cartridge will move as well in order to print a continuous circuit trace.

Within the software, there will be functions that are responsible for determining when the nozzles will trigger. For example, some pins will turn on, while other will be disabled in order to enable only a specific set of nozzles. This will ensure accuracy in the circuit trace. This software design is paramount to the overall design of the Universal Circuit Fabricator, as the inkjet cartridge actually holds the conductive ink that will print to the substrate.

After printing has been completed, there will be a final check signal that will be sent to the inkjet control system that will determine that the design has completely been received. After the inkjet system has determined that the design is complete, the inkjet nozzles will be disabled and will return to their initial configurations.

6.3.2 Variable Descriptions

This section describes the main variables present within the inkjet cartridge control system code in order to perform its main functions. These variables will be used within the subsystem code itself. The Table below lists the variables and a brief description of their use.

Table 6 – Variable Descriptions for the Inkjet Cartridge Control System

Variable Name	Type	Description
Printhead_target_reached	Boolean	Once this variable has been set to 1, it means that the circuit trace has finished transmitting, turning off the print head inkjet nozzles.
nozzle_count	int	The number of nozzles that are present in the inkjet cartridge. Also used for declaring what nozzles to turn on during some functions during the printing process
nozzle_state	Boolean	Determines whether a specific nozzle is on or off. Used in the nozzle_set function to determine if the nozzle is printing or not
nozzle_active[n]	Boolean	An array based on the number of nozzles in the printhead that determines whether the nozzle is on or off. Used for triggering specific nozzles on the inkjet.
line_code	byte	Determines if a line is a nozzle code or a move code. Either a value of 0, 1, or 2. Used for both the inkjet system, the motor control system, and the data processing system to determine what is being read from the bitmap

6.3.3 Function Descriptions

The table below describes the main functions within the inkjet cartridge control system that will be used for the inkjet print head. The table will list the relevant functions for the subsystem, their input and output variables, and a brief description of their use

Table 7 – Function Descriptions for the Inkjet Cartridge Control System

Function Name	Inputs	Outputs	Description
nozzle_trigger()	(void)	(void)	Triggers all print head nozzles as on in order to print the circuit trace
nozzle_set()	nozzle_state	(void)	Determines what nozzles to turn on or off when the state of the nozzle is given
delayMicroseconds()	int	(void)	Creates a delay in the code that will pause the program in order to ensure that the inkjet cartridge nozzles don't spray continuously creating inaccurate traces

6.4 MOTOR CONTROL SYSTEM

Within the following section, the software design for the motor control system will be explored. As stated in the hardware section above, the Universal Circuit Fabricator will make use of two stepper motors within the final design that will be connected to a subsystem on the final design's printed circuit board. This subsystem is fundamental to the overall design of the UCF because the motors will move the gantry and ribbon system that holds the inkjet system.

6.4.1 Software Configuration

The software configuration of the motor control system within the UCF has a main primary purpose: controlling the x-axis and y-axis stepper motors within the printer frame. This is a fundamental system because the motors will be connected to the print head nozzle itself by a gantry and ribbon system that will allow the print head to move in the x and the y direction. As the motors move the

ribbons, the print head will move over the substrate and print the conductive ink that will form a final conductive continuous trace in the glass substrate.

In order to do this, the software must accommodate instructions from the microprocessor that, in turn will be received from the data collection and processing system. The microprocessor within the final PCB must process the data that it receives, and translate that parsed data into simple character instructions to the stepper motor control system. This will allow the motor control system to recognize whether it should move in the x direction, y direction, or not move at all.

Because movement is perhaps one of the most important facets of our design, the code must be efficient and retain a high accuracy. The code must also read from the buffer in the processor itself so the speed of the print is not interrupted. Finally, the inputted circuit trace bitmap that is given originally to the microprocessor must be processed and translated to a variable that the motor system will easily recognize. Within the code, the design group must parse the raw data and translate it into a character that the processor will read one by one in order to tell the motors to move.

Below is a visual block diagram of the motor control system and how the microprocessor will give movement instructions to the motor control system. The below diagram has the main variable that the data collection subsystem will output to the microprocessor (PackagedData). The microprocessor will convert that variable into a char data called line_pos that will be read by the motor control system. This variable will also be used in the Data Collection & Processing System and the Inkjet Cartridge Control System.

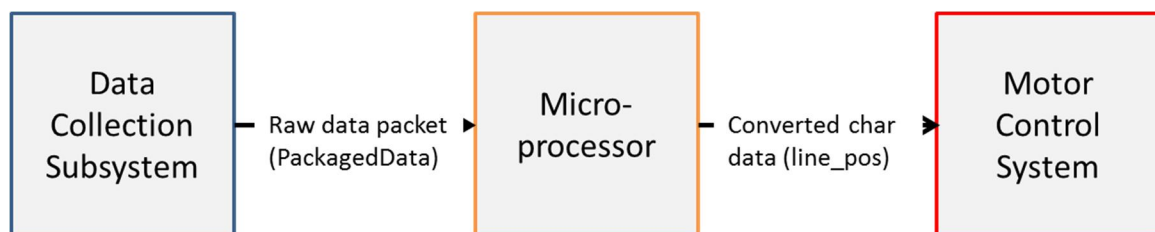


Figure 11 – Motor Control System Data Block Diagram

6.4.2 Variable Descriptions

The figure below lists the main variables that are important to the motor control system for the Universal Circuit Fabricator. As stated above, perhaps the most important variable within the table below is the line_pos variable, which determines whether the coordinate being read by the motor system will be a movement in the x direction, the y direction, or have no movement. The other variables are important in keeping track of the general coordinate movement of the motors, as well as resetting position after the printing is done.

Table 8 – Variable Descriptions for the Motor Control System

Variable Name	Type	Description
line_pos	byte	The data that will determine where the line of the bitmap is being read. Either an X-axis movement, Y-axis movement, or to turn on the nozzles. Used by the inkjet cartridge control system and the motor control system
Target_reached	Boolean	When the printing target is reached for either the x or y direction, the motors will stop all movement and reset
X_enabled	Boolean	Represents whether or not the x-axis stepper motor is turned on or disabled
Y_enabled	boolean	Represented whether or not the y-axis stepper motor is turned on or disabled
X_steps_per_mm	Int	Determines the constant axes steps the x-axis stepper motor needs to move 1 millimeter.
Y_steps_per_mm	Int	Determines the constant axes steps the y-axis stepper motor needs to move 1 millimeter.
Invert_x	Boolean	Determines whether or not to invert the movement of the x-axis stepper motor
Invert_y	Boolean	Determines whether or not to invert the movement of the y-axis stepper motor

6.4.3 Function Descriptions

Within the following table, the main functions and subroutines for the motor control system are listed for the Universal Circuit Fabricator. In the table is the function name, their input and output variables necessary for operation as well as a brief description of what the function does.

Table 9 – Function Descriptions for the Motor Control System

Function Name	Inputs	Outputs	Description
Enable_all()	(void)	(void)	Enables both the x-axis and the y-axis motors using the variables x_enabled and y_enabled. Can also be used to turn a single motor on
Set_x_direction()	Tempdir	(void)	Sets the direction of the x-axis motor when using movement.
Set_y_direction()	Tempdir	(void)	Sets the direction of the y-axis motor when using movement
Step_x()	(void)	(void)	If the x-axis stepper motor is enabled, the function creates a single step in the x-axis.
Step_y()	(void)	(void)	If the y-axis stepper motor is enabled, the function creates a single step in the y-axis.
X_start()	(void)	(void)	Uses the step_x function to have a continuous movement in the x direction through constant step motor movement. Vice versa for movement in the y-axis
Y_start	(void)	(void)	Uses the step_y function to have a continuous movement in the y direction through constant step motor movement. Vice versa for movement in the x-axis

6.5 ANNEALING SYSTEM

Within the following section, the Annealing System software design will be discussed for the UCF, which includes the heating print bed, as well as thermocouple sensors that would be present in the design to determine the temperature. The sensors must recognize a temperature limit and give an effective measurement in order to ensure that the glass substrate is properly heated for the silver nitrate ink to fuse and provide a continuous conductance.

6.5.1 Software Configuration

The Annealing System only has one main primary purpose, which is to maintain a 90 degree Celsius on the heated print bed. When the annealing system starts, the microcontroller will turn on the hot plate and begin monitoring the substrate via the thermocouples in the initial and final design. When the thermocouple reaches an acceptable temperature, it must maintain that to create an optimal annealing process for a defined time period. Therefore, the code must have a subroutine that determines whether the temperature is hot enough before controlling the hot plate so it does not get hotter.

6.5.2 Variable Descriptions

The figure below lists the main variables that are important to the annealing system for the Universal Circuit Fabricator. From above, the two integer variables below are dependent on the thermocouples linked to the heated print bed. The sensor_temperature will be the given measurement of the temperature of the print bed, which will be taken through the use of the thermocouple that will be on the heated print bed and connected to the final PCB design. The set_temperature value is even more important, as it is the temperature that the print bed must always strive to be in order for the conductive ink to solidify and anneal and provide a continuous trace.

Table 10 – Variable Descriptions for the Annealing System

Variable Name	Type	Description
Sensor_temperature	Int	The given temperature that the thermocouple is measuring of the print bed. This variable is used to keep track of the print bed temperature to keep it at an optimal level
Set_temperature	Int	The optimal temperature that will act as a limit for the thermocouple. The sensor_temperature must not go above this level in order to prevent overheating

6.5.3 Function Descriptions

Within the following table, the main functions and subroutines for the annealing system are listed for the Universal Circuit Fabricator. In the table is the function name, their input and output variables necessary for operation as well as a brief description of what the function does. Within the annealing system, the main function will be to read the temperature in the thermocouples in order to compare it to the optimal temperature that the heated print bed will operate on. The secondary function will be to update the temperature, where they take the outputted sensor_temperature and if it does not match the optimal set_temperature, it will turn on or off the heaters until the goal temperature is reached.

Table 11 – Function Descriptions for the Annealing System

Function Name	Inputs	Outputs	Description
Read_temperature()	(void)	Sensor_temperature	Main function that reads the temperature of the hot pate through the thermocouples and compares it to the set_temperature
Update_temperature()	(void)	(void)	Reads the thermocouple sensors from read_temperature, if doesn't match set_temperature, turns on or off heaters

6.6 DATA COLLECTION & PROCESSING SYSTEM

Within the following section, the Data Collection & Processing Subsystem of the Universal Circuit Fabricator will be discussed, which includes the SD card reader that will store basic circuit traces, as well as the capacitive touch buttons and microprocessor as they will write data to the processor for further use.

6.6.1 Software Configuration

The Data Collection & Processing Subsystem within the UCF has multiple purposes. First, it must collect data from the Capacitive Touch Screen input from the user. This data which is originally simple button presses must be collected and processed into a series of movement in the UCF that will draw a basic circuit trace. From the capacitive touch screen, the data will move into the ATmega328P microprocessor, processed, and move into the other subsystems.

This subsystem must also be responsible for implementing the SD card reader that will be present within both our initial and final design. A circuit design bitmap or jpeg must be able to be read by the SD card reader and transferred to the microprocessor for further processing. Subsequently, circuit traces will be stored in the buffer of the microprocessor so that the data will be used further on in the process by the motor, inkjet, and annealing subsystems so that a final circuit trace is printed.

The software for this subsystem is important because if the circuit trace image cannot be read by the UCF, then the circuit will not be able to print a legible trace. This phase of software design drives the main system level software. As long as an SD card is inputted and being read, the data collector and processor will continuously run. The code must be extremely efficient so a bottle neck is not created.

Initially, the data packets will be acquired from either the user input capacitive touch screen or the SD card reader as stated above. Here, the microcontroller code will strip down the packet of data that is being read and convert it into utilizable and legible data for the microprocessor. Initially, the packet will contain the bitmap image of the circuit trace, as well as the coordinates of the initial position of the bitmap. The data processor will take this data and create a further object from it that the rest of the UCF will utilize.

An important check for this processing phase is to ensure that the data remains organized and does not change as it moves along the system. If the processor does not keep the integrity of the data packet, then the circuit trace image could become corrupted and might create an inaccurate circuit trace on the packet. Finally, as the data has been converted and processed, it must move this now usable data out of the processor and into the other sections of the software design. The data will be converted into inkjet data, motor control data, and annealing system data and work simultaneously to print a legible circuit trace on the substrate.

Below is a logical block diagram view of the software for data collection and processing for the Universal Circuit Fabricator

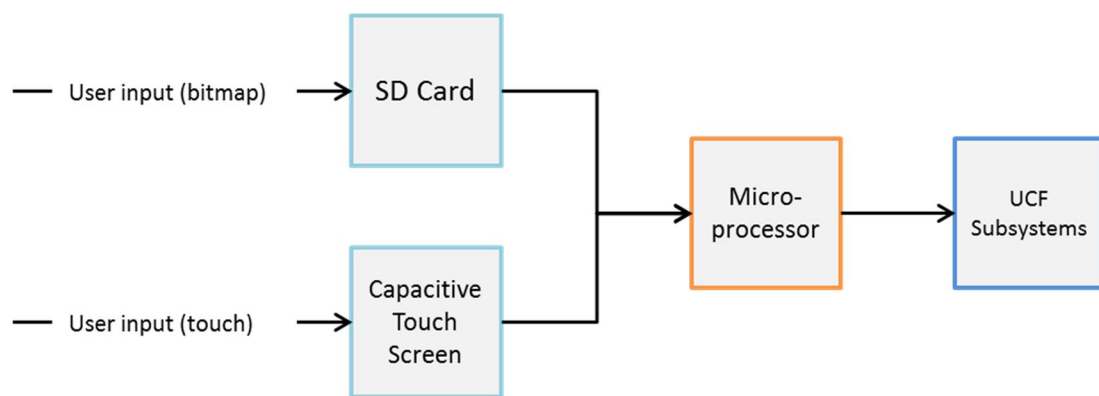


Figure 12 – Data Collection and Processing Block Diagram

6.6.2 Variable Descriptions

Below lists the variables that are most important towards the Data Collection and Processing Subsystem. As seen below, perhaps most important in this subsystem is the buffer, which keeps track of the code read from the SD card, temporarily storing it. The packagedData from the SD card must be converted by use of a function to be read by the other subsystems in the UCF.

Table 12 – Variable Descriptions for the Data Collection & Processing System

Variable Name	Type	Description
buffer_size	int	The buffer is where the code read from the SD card will be temporarily stored before printing. During normal printing and movement, the buffer will be used to read the bitmap image to be printed as a trace
packagedData	String	Data that has been collected from the SD card to be further converted and processed by the microprocessor
max_offset	int	Keeps track of the offset maximum allowed in the look ahead of the buffer. Makes sure that if exceeded, the buffer will not look at the next line.
temp_block[n]	boolean	The variable used to convert the packagedData to a character that will be stored in an array. The array will determine whether or not data is present in the bitmap, representing to print or not to print
line_pos	byte	The data that will determine where the line of the bitmap is being read. Either an X-axis movement, Y-axis movement, or to turn on the nozzles. Used by the inkjet cartridge control system and the motor control system

6.6.3 Function Descriptions

The following table describes and lists the main functions that will be used in the Data Collection and Processing Subsystem. These functions are important because they allow the UCF to read from the SD card, as well as a function to convert the packagedData from the SD card into a line position of the bitmap so the other subsystems like the motor control and the inkjet print head system knows whether to move or print.

Table 13 – Function Descriptions for the Data Collection & Processing System

Function Name	Inputs	Outputs	Description
read_sd_card()	(void)	(void)	Responsible for reading the bitmap file from the SD card and processing the .bmp format to data stored in the buffers for printing. Reads one character at a time until line is processed
write_to_buffer()	(void)	(void)	After reading the data, the data is converted by character and written to the buffer for temporary storage
buffer_next()	(void)	(void)	Allows movement to the next memory slot in the buffer. Used in the write_to_buffer() function
buffer_look()	(void)	(void)	Looks to the next buffer slot and determines whether a character is present, determining if buffer_next() is used.
buffer_update()	(void)	(void)	Updates the buffer when new bitmap is read from SD card. Uses the write_to_buffer() function. Also clears the buffer before next print

6.7 GRAPHICAL USER INTERFACE

The Graphical User Interface allows the user to interact with the Universal Circuit Fabricator. The GUI software is responsible for presenting the menu structure that allows options for the user to design and print circuits.

6.7.1 Software Detail & Configuration

The Graphical User Interface is the communication medium between the user and the Universal Circuit Fabricator's software. The GUI allows the user to select a circuit design off of an SD card, or draw a custom circuit using the capacitive touch panel. It shows a preview of the printed circuit prior to printing. Print progress and ink level are displayed during the printing process. A temperature gauge and timer are displayed during the annealing process.

6.7.1.1 Top Menu

The Top Menu is the first screen that the user sees upon restarting the software. It gives a choice between printing a design off of an SD Card or from a Touch Screen input. This is also the screen that the software will display after either the user selects cancels or selects main menu at any time.

6.7.1.2 SD Card Root Navigation

The SD Root Navigation screen is displayed when the user selects SD Card from the Top Menu. It allows the user to browse the root folder of an SD card to find a pre-designed bitmap of a circuit trace for printing. The user can scroll up and down and also go up and down folder levels. The user can either select a design for printing or cancel back to the Top Menu.

6.7.1.3 Touch Screen Live Preview

The Touch Screen Live Preview screen is displayed when the user selects Touch Screen from the Top Menu. It allows the user to draw a circuit using the capacitive sensor input device. The cancel button returns the user to the Top Menu. The clear screen button clears the live preview and allows the user to start the design over. The print button takes the user to the print preview screen.

6.7.1.4 SD Card / Touch Screen Print Preview

Both the SD Card and Touch Screen Print Preview screens display a preview of what the printed circuit will look like. The user can either press the cancel button to return to the Top Menu or press the Print button to begin the printing process.

6.7.1.5 Print in Progress

The Print in Progress screen is displayed while the conductive ink is being deposited onto the substrate. It displays a preview of the printed design as well as a progress bar and ink level percentage. There is also a Cancel Print button that allows the user to return to the Top Menu

6.7.1.6 Ready to Anneal?

Once the printing process has completed, the Ready to Anneal? screen is displayed before starting the annealing process. This allows the user to inspect the deposited ink on the substrate and verify that the design will function properly once the ink has been annealed to a permanent state. The user can either press the cancel button to return to the Top Menu or press the Yes button to start the annealing process.

6.7.1.7 Annealing in Process

The Annealing in Process screen is shown while the hot plate is on and the substrate is being heated. The screen displays a temperature gauge that displays the reading from the thermocouple. A timer is also displayed that counts down the amount of time needed to complete the annealing process.

6.7.1.8 Print Summary

The Print Summary screen is displayed once the timer on the Annealing in Process screen has expired. It displays the time elapsed from the start of the printing process to the end of the annealing process. A Main Menu button is displayed to return the user to the Top Menu.

6.7.2 GUI Code Composer

The Graphics User Interface will be developed in a program called GUI Code Composer. This software was developed by Texas Instrument, and was designed as an addition to Texas Instruments' already famous Code Composer Studio. It was designed to give the user easy to use features to create a graphics user interface for their projects.

The program works by designating figures with different commands. For example, the program gives you diagram of a button, and you can place it within an allotted space. The user of the program then assigns a variable name to that button, which is used to receive user input. This is only a simply case of what the Graphic User Interface Code Composer can use.

For the purposes of this project the group will be using the GUI Code Composer to design the GUI for the Universal Circuit Fabricator. This GUI will be developed and designed within this program. After the design is completed, GUI Code Composer uses a set of functions and algorithms to create a source code. This source code is exported automatically to generate a DSS script that performs system initialization tasks. This source code is then flashed to the development board of the group's desire.

Within GUI Code Composer there is a powerful debugger, to make sure that the diagrams that you are creating have the correct labeling, in order to reduce programming time. In addition to verifying that the code executes properly, the

debugger also is enabled to halt the CPU before making a target access. This allows for greatest compatibility across various devices.

6.7.3 Capacitive Touch Input

The capacitive touch sensor will be used to navigate the graphic user interface. This sensor contains five available tactile buttons that will be used to navigate the menus. Each button will be assigned a variable name that the code will interpret and execute different commands. Within the capacitive touch sensor there are also two LEDs that indicate that one of the tactile buttons has been activated. The table below describes the variables used to program the capacitive touch input.

Table 14 - Capacitive Touch Sensor Variable Assignment

Variable Name	Type	Description
touch_button1	int	The button 1 located on the capacitive glass plate. The signal received by the microprocessor will either be a value of 0 or 1. The value zero is that the button has not been activated. The value of one is that the button has been activated.
touch_button2	int	The button 2 located on the capacitive glass plate. The signal received by the microprocessor will either be a value of 0 or 1. The value zero is that the button has not been activated. The value of one is that the button has been activated.
touch_button3	int	The button 3 located on the capacitive glass plate. The signal received by the microprocessor will either be a value of 0 or 1. The value zero is that the button has not been activated. The value of one is that the button has been activated.
CTS_LED1	int	One of the LEDs located on the capacitive touch sensor. This indicates that a button has been pressed. The microprocessor sends a value of one to turn on the LED to give the user feedback.
CTS_LED2	int	One of the LEDs located on the capacitive touch sensor. This indicates that a button has been pressed. The microprocessor sends a value of one to turn on the LED to give the user feedback.

6.7.4 LCD Display

The LCD Display will be used to display the graphic user interface. The LCD Display has touchscreen capabilities, which would be a great feature to have in any project. But for the purposes of simplicity and ease of use, the group will be using a capacitive touch sensor to navigate the graphical user interface. The software for this Display will be developed in GUI Code Composer. Then it will be uploaded to the board that the project will be using.

Within each sub-menu there will be different selections that will allow the user to navigate to the desired sub-menu. In order to navigate these, there will be a separate program that will run the graphics user interface and be able to take input from the user. This program will work in conjunction with the capacitive touch sensor to navigate the sub-menus. The table below describes the functions used in the programming of the LCD display.

Table 15 - Function Description for GUI System

Function Name	Inputs	Outputs	Description
right_oper()	(touch_sensor3)	(void)	Responsible for telling the graphic user interface program to move the cursor to the next element in the menu
left_oper()	(touch_sensor1)	(void)	Responsible for telling the graphic user interface program to move the cursor to the next element in the menu
up_oper()	(touch_sensor5)	(void)	Responsible for telling the graphic user interface program to move the cursor to the next element in the menu
down_oper()	(touch_sensor4)	(void)	Responsible for telling the graphic user interface program to move the cursor to the next element in the menu
enter_oper()	(touch_sensor2)	(void)	Responsible for telling the graphic user interface program to select the element that the cursor is currently on

6.7.5 Block Diagram

Figure 13 shows a visual representation of the graphical user interface. The diagram represents what the user will see on the LCD Display as well as the flow from between each menu screen. The menu options are explored with user input on the capacitive touch sensor.

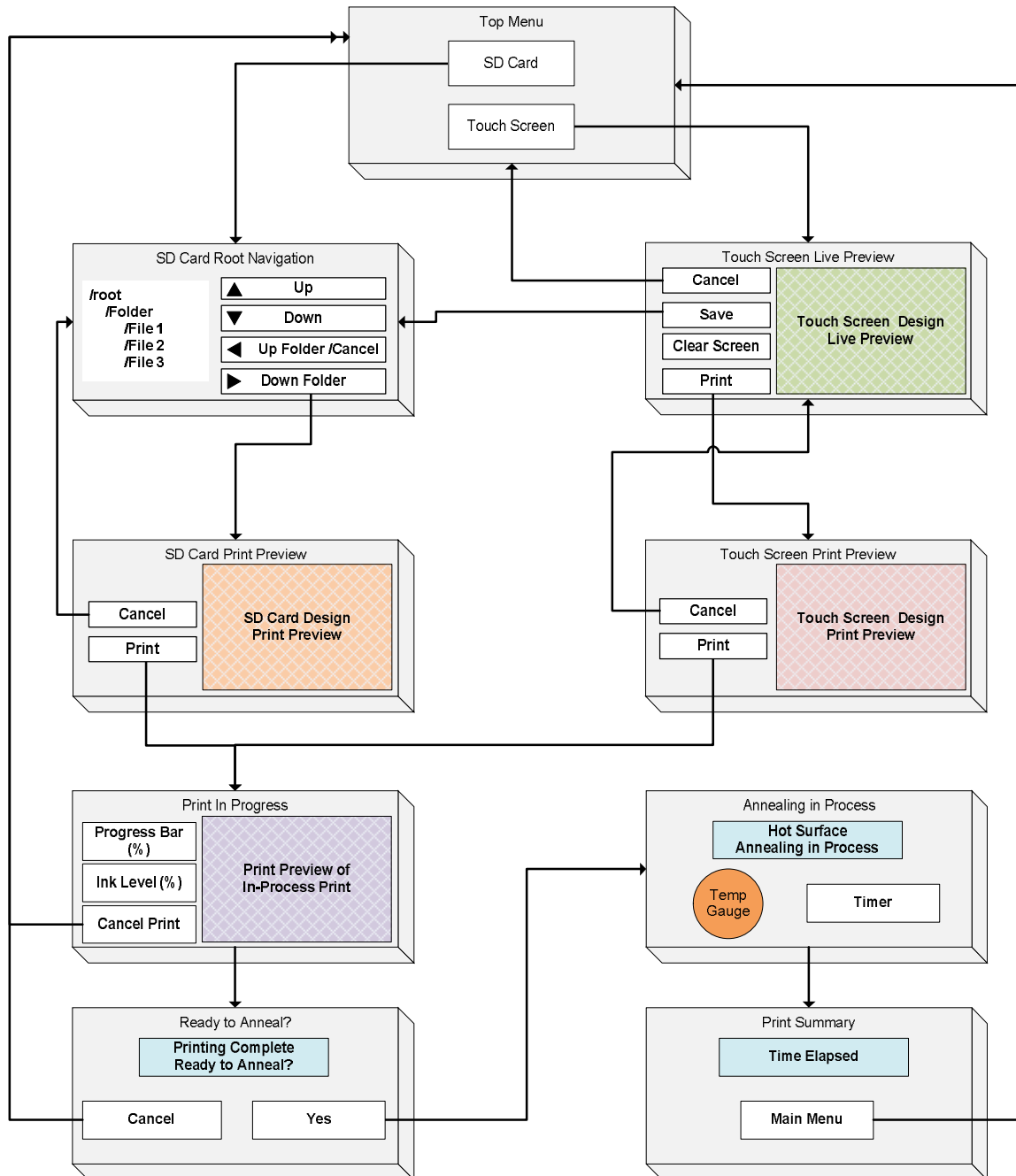


Figure 13 Visual Representation of Graphical User Interface

7 Prototype Design Summary

The Universal Circuit Fabricator is designed as a complete system that is operated by the user using the graphical user interface and capacitive touch sensor. The user is able to operate the UCF with full functionality without the need for an external computer (with the exception of providing an external design via SD Card) or any other kind of hardware.

In order to create an end product system that provides an intuitive user experience, the importance of successfully integrating the subsystems cannot be understated. Figure 14 describes the spatial layout of the subsystems and how they will function together to deliver the specifications required for a functioning product.

7.1 INPUT SUBSYSTEMS

The Input Subsystems, shown in green color in figure 14, are designed to be used by the user to navigate the menu options of the embedded microcontroller and in turn, control the processing and output subsystems. In addition, the input subsystems are designed to input a design file for use by the embedded microcontroller to print a user designed conductive ink trace.

7.2 PROCESSING SUBSYSTEM

The Processing Subsystem, shown in yellow color in figure 14, is designed to run the software that supports and powers the various subsystems. It analyzes the user's input from the input subsystems and presents the graphical user interface for user input. The processing subsystem is also designed to control the various printing subsystems to successfully print a design.

7.3 PRINTING SUBSYSTEMS

The Printing Subsystems, shown in shades of blue in figure 14, are designed to work together to move the ink jet print head, deposit conductive ink from the print head, and heat the trace to anneal the ink. The X and Y axis motors are designed to move the print head in 2 dimensions based on the control of the processing subsystem. The print head is designed to deposit conductive ink onto the substrate based on the control of the processing subsystem. The heated print bed is designed to heat to annealing temperatures after printing has been completed to finalize the printed design.

7.4 OUTPUT SUBSYSTEM

The output subsystem, shown in orange color in figure 14, is designed to display the graphical user interface that is run in the processing subsystem. It provides the user with a feedback loop when drawing a custom design and also displays options and statistics of print and system status.

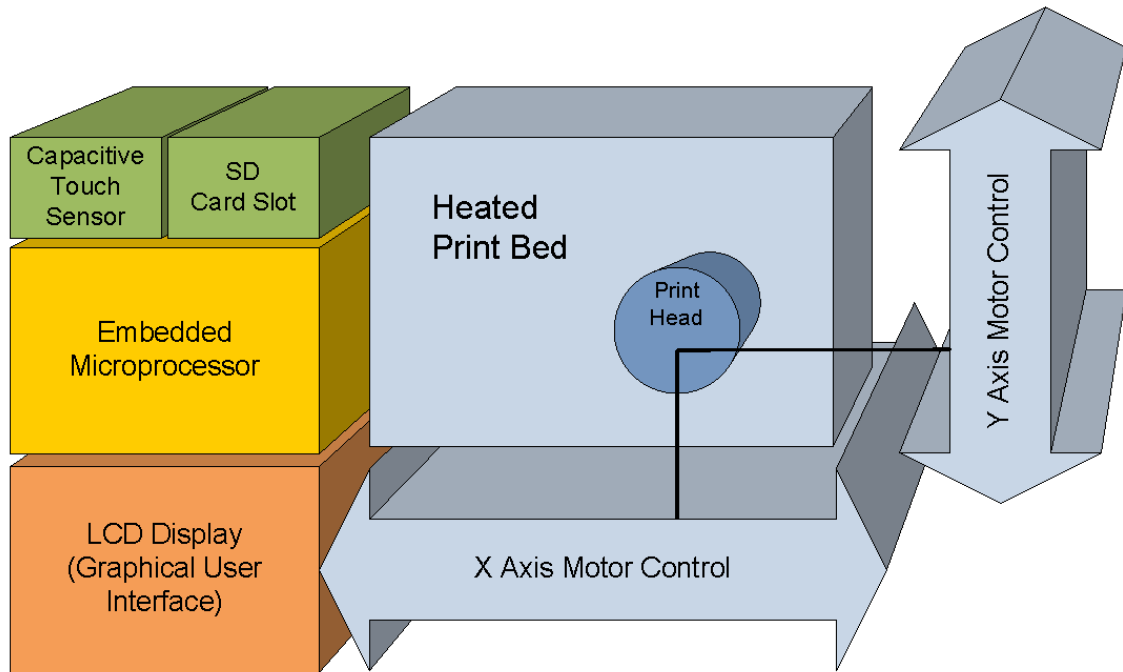


Figure 14 Final Design Physical Layout

8 Build Plan

The following sections will describe in details the process that the group will take in order to create a prototype, for the Universal Circuit Fabricator. It will also include the components that will be used in the prototype, with a brief description of each component. There will also be a section on software development, which will include the different environments that will be used to program the prototype for Universal Circuit Fabricator.

Finally, a section within the Build Plan will also go into the design of the final printed circuit board, with logical pin assignments and input output relationships between the different subsystems and the microcontroller. Both pin assignment tables and a figure of the final design will be included in the following sections.

8.1 PROTOTYPE DESIGN

During the prototype phases of the project, several products will be used as tools for prototyping. By building prototype systems, the detailed design work can be accomplished more cost effectively in less time. For example, creating custom PCBs each time a connection needs to be fixed would be incredibly expensive. Additionally, using the proper tools for software development can greatly decrease coding and debugging time.

8.1.1 Hardware Development

The main component in the system is the microcontroller development board. There are two possible options to this component, either and MSP430 or an Arduino Uno. After thorough analysis and comparison, the group has decided to select the Atmel ATmega328P as the heart of the Universal Circuit Fabricator.

The board utilized is designed by Arduino, who also provide the software libraries for the microprocessor. The microcontroller which can be seen in the figure below features multiple I/O ports, and a mini USB for programming. This version of the Arduino UNO microprocessor has an advantage over the rest, which is that it has a larger capacity for memory and the flash memory is enough for the purposes that is needed. This board has 10 I/O ports which is much larger than needed

This development board will be used to prototype the Universal Circuit Fabricator, before it is placed on a PCB. Since the Arduino UNO is a development board there is plenty of support from the community as well as from TI. Figure 15 is a layout schematic of the Atmel ATmega328P microprocessor.

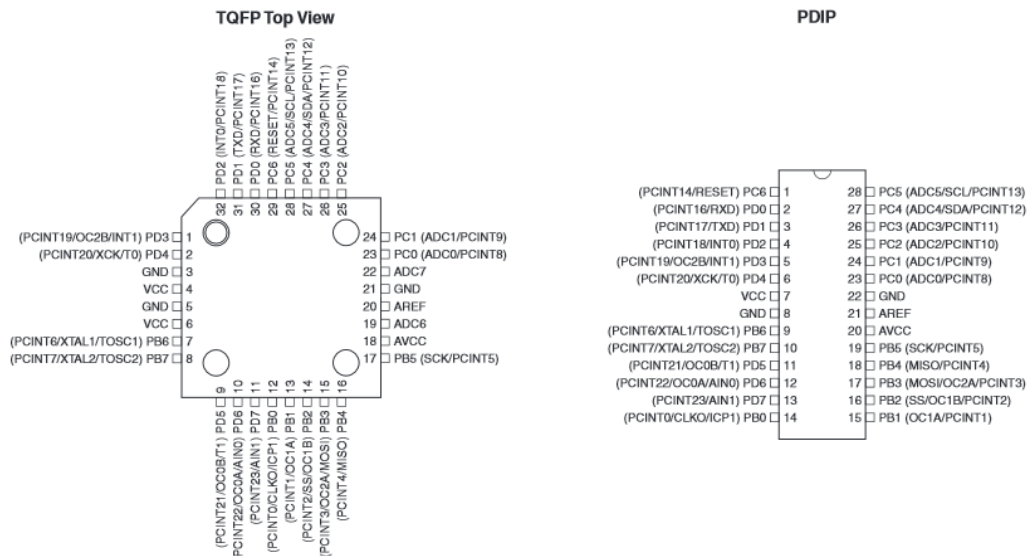


Table 16 describes the specifications and features of the Arduino UNO prototyping microcontroller board. This board meets the design requirements for the UCF hardware specifications.

Table 16 – Arduino UNO Features

Arduino UNO	
Clock Speed	16 MHz
Memory	32 KB Flash
Onboard Memory(RAM)	2 kB
A-D Converter	12-bit
Digital I/O Pins	14

The SD card breakout board can be seen in the figure below. The breakout board contains the SD breakout board PCB, and the SD card push/pull socket. These two segments must be soldered together, as well as the jumper pins. This board will be used to read an SD card the user will fill with schematic diagrams. Figure 16 is an image of the SD Card breakout board that will be used in the UCF design.



Figure 16 – SD Card Breakout Board (permission pending from 43oh.com)

The InkShield breakout board can be seen in the figure below. This board will be used to control the inkjet cartridge that will contain conductive ink to trace circuit designs. After careful consideration the group found that this was the only option to utilize a printer inkjet cartridge. There is no other open source inkjet cartridge driver that can be used for development and prototyping.

The InkShield is powered by the power output pins on the development board. It requires at least 9 volts from the input port; there is a step up switching regulator

built in the board to bring the voltage up to 20 volts. The required amount to power the inkjet cartridge is stated in the specifications sheet, which is 20 volts in order for it to be operational.

There are twelve nozzles on the HP C6602 Inkjet Cartridge, and in order to not use twelve pins from the Dev-board. The InkShield uses a CD4067 de-multiplexer to drive two ULN2803 Darlington arrays; this reduces the pins used to four instead of twelve. Figure 17 is an image of the assembled InkShield breakout board.

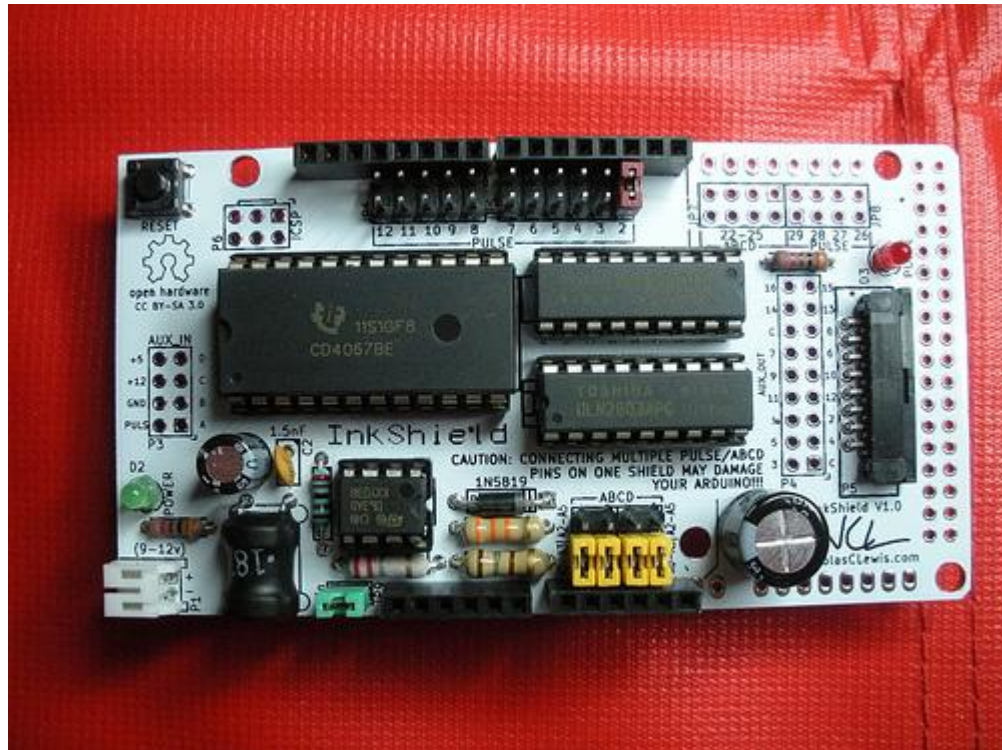


Figure 17 – InkShield Breakout Board (permission pending from Nicholas C Lewis)

The schematic diagram below in figure 18 is for the InkShield. It consists of the circuit traces and the pin layout within the InkShield Breakout board. It shows the pin locations for each component, as well as the connection between each component.

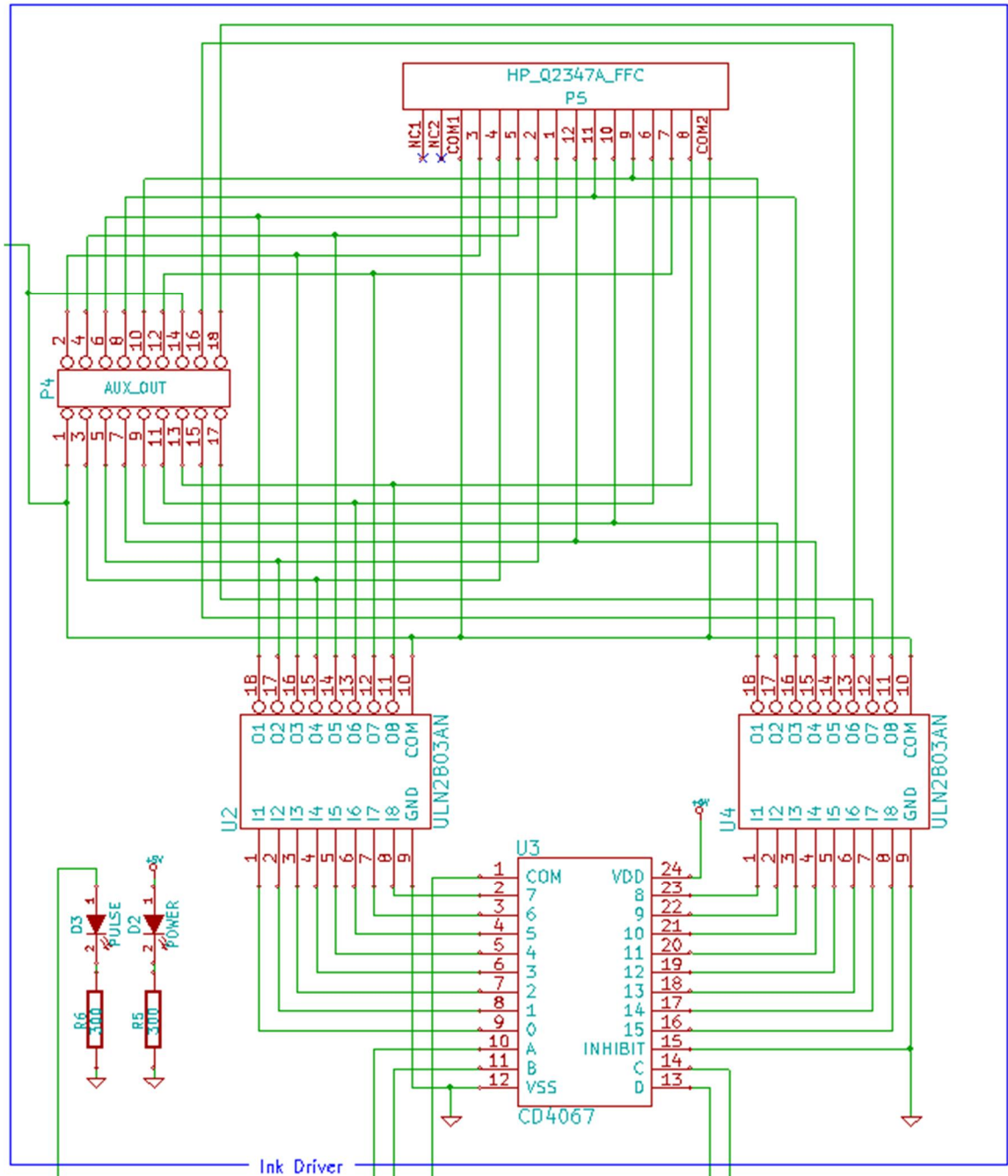


Figure 18 – Inkshield Layout Schematic (permission pending from Nicholas C Lewis)

The motor control system will consist of two stepper motors (ROB-10846) and a motor control breakout board. There were two possible options for the motor selection, it was between stepper motors and servo motors. The main difference between servo and stepper motors is the amount of poles within the stator cylinders. A servo motor can have from 4 to 12 possible stator poles, unlike a stepper motor which can have from 50 to 100 poles. The main draw from this information is that stepper motors have precise control, and the servo motors have more torque. Therefore since the Universal Circuit Fabricator needs precision, the group has decided that it will use the NEMA 17 stepper motors. The motor has the following specifications:

Table 17 - Stepper Motor Specifications

Stepper Motor (NEMA 17)
Step Angle (degrees) : 0.9
2-Phase
Rated Voltage: 3V
Rated Current: 1.7A/phase
5mm Diameter Drive Shaft
Holding torque: 48 N*cm
400 steps/rev

The motor control breakout board (BOOST-DRV8711) is shown in the figure below. This board is able to drive a bipolar stepper motor, with the following ratings 8-52V, 4.5A. Since this board is compatible with the Launchpad board that the project will be using, programming and development will be much easier. This breakout board has the schematic diagram, which contains the pin allocation and chip placement, shown in the figure below.

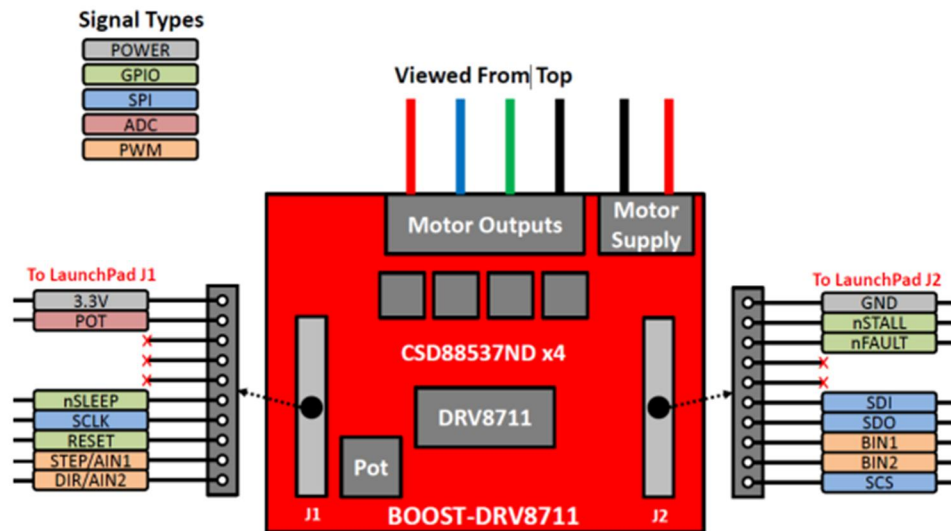


Figure 19 – BOOST-DRV87411 Motor Control Board (with permission from Texas Instruments)

The LCD display (EB-LM4F120-L35) that will be used in the prototype is shown in the figure below. This display is manufactured by Kentec Display; they have provided the community with information, which will be used for development purposes. This LCD will be used to display the Graphic User Interface that the Universal Circuit Fabricator will utilize. The display is approximately 3.5 inches, more than an appropriated size for this printer's uses. It will operate with the 3.3V that the Launchpad can provide.

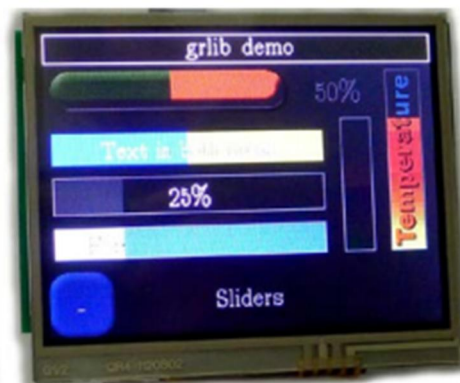


Figure 20 – LCD Display (permission pending from Kentec Display)

The following table shows the pin sockets that will be used by this display: (Provided by Kentec Display).

Table 18 – LCD Display pin socket description

Pin	Symbol	AtMega Pin	Description
1	3v3	+3.3V	Power Supply
2	LCD_D5	PB5	Data bit 5 for LCD
3	LCD_D5	PB0	Data bit 0 for LCD
4	LCD_D5	PB1	Data bit 1 for LCD
5	TOUCH_XP	PE4	Screen Terminal (left)
6	TOUCH_YP	PE5	Screen Terminal (top)
7	LCD_D4	PB4	Data bit 4 for LCD
8	LCD_WR	PA5	Write control signal
9	LCD_RS	PA6	Register/Data select for LCD
10	LCD_CS	PA7	Chip select for LCD

The capacitive touch breakout board that will be used in the prototype phase is shown in the figure below. This kit is composed of an ITO glass capacitive touch sensor and a connector board. This kit is developed and manufactured by Kentec Display; they have provided the community with information, which will be used for development purposes. This capacitive touch will be used to collect the input of the user, when he/she is navigating the user interface. The ITO glass capacitive touch sensor is 3.5 inches diagonally; it also contains 5 function keys.

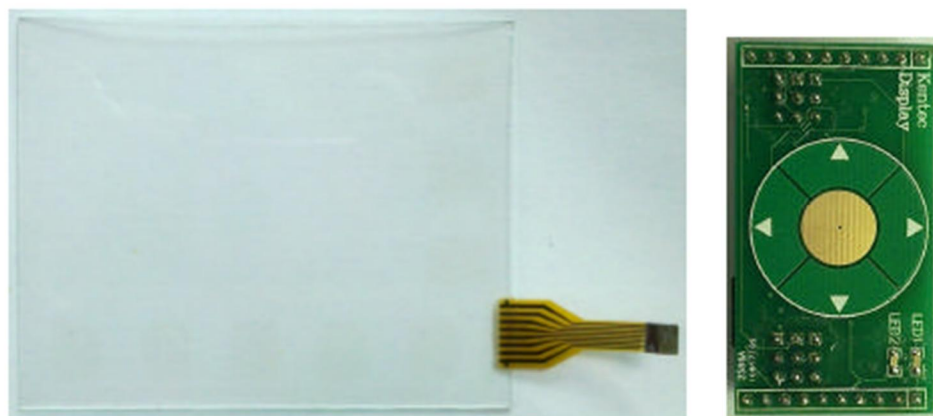


Figure 21 – ITO Glass Capacitive Touch Sensor (permission pending from Kentec Display)

The following diagram demonstrates the button layout of the capacitive touch sensor. It shows the five buttons that are available for use, as well as the location of each one.

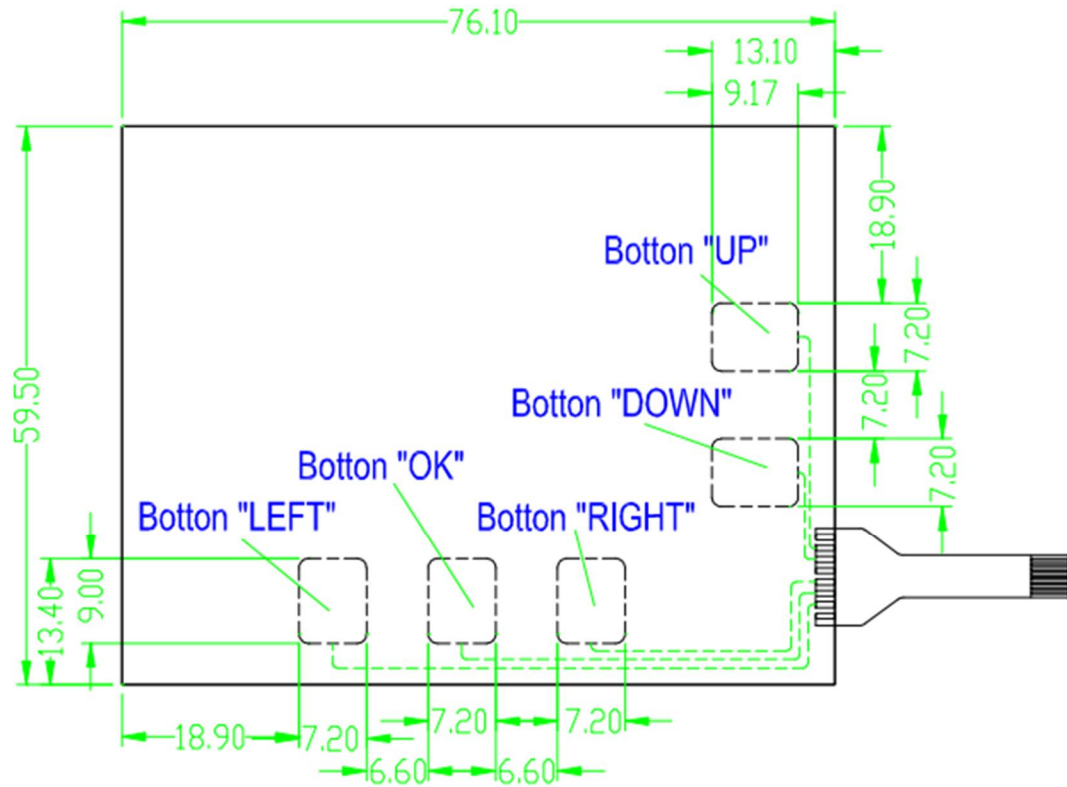


Figure 22 – Capacitive Touch Sensor Button Layout (permission pending from Kentec Display)

The following table shows the pin sockets that will be used by this capacitive touch sensor: (Provided by Kentec Display).

Table 19 - Capacitive Touch Sensor pin socket description

Pin	ATMega Port	Signal	Description
1	VCC	VCC	Supply Voltage
2	P1.0	TACLK_Input	Timer0_A clock signal input
3	P1.1	CA_Ref	Comparator reference
4	P1.2	S1	Touch-Sense 1
5	P1.3	S2	Touch-Sense 2
6	P1.4	S3	Touch-Sense 3
7	P1.5	S4	Touch-Sense 4
8	P1.6	S5	Touch-Sense 5
9	P1.7	CA_OUT	Comparator output
10	RST/SBWTIO	NC	Reset line for SBW JTAG data, not connected to sensor
11	TEST/SBWTCK	CN	Test line for SBW JTAG clock, not connected to sensor
12	P2.6/XOUT	LED1	LED1 positive drive
13	P2.7/XIN	LED2	LED2 positive drive
14	GND	GND	Supply ground

8.1.2 Software Development

As for the software development, there is one specific development environment that the group will be using that is the Arduino IDE from Texas Instruments. It is based off the C language; therefore, libraries that can be used in Arduino can also be used in many programs. The original prototype utilizes the Arduino IDE, because of the Arduino UNO development board that was utilized to run and test the InkShield. The code is written in C/C++ language so that it can be used in both environments. The following figure is a screenshot of the Arduino IDE.

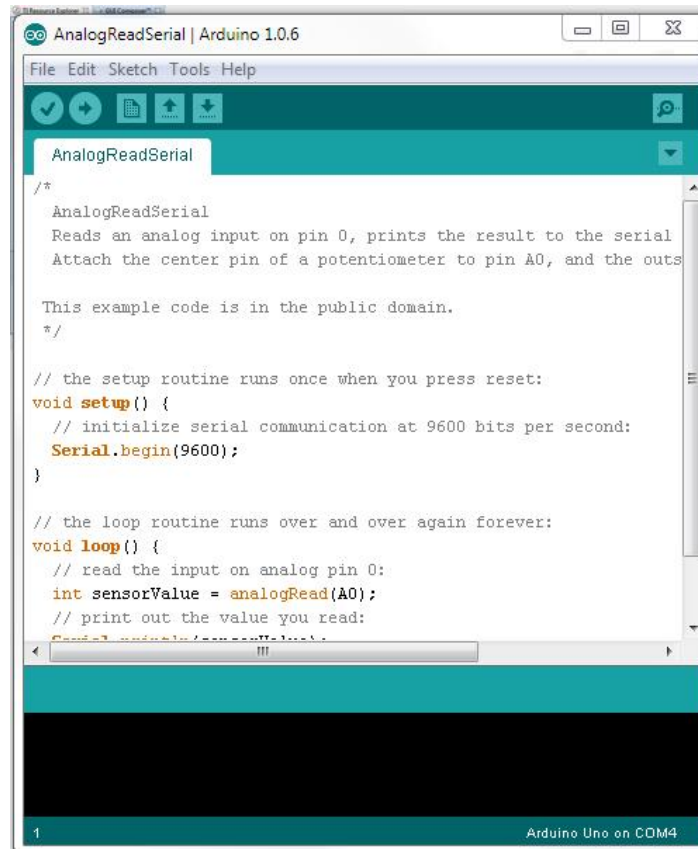


Figure 23 – Arduino IDE (with permission from Arduino)

Data Collections and processing will be written in C code and will be developed in Code Composer Studio, by Texas Instruments. This program allows the group to design and test the code utilized by the Universal Circuit Fabricator. In addition, the code completion feature saves countless hours by suggesting the function or data you were attempting to refer to on the fly while you code. Perhaps the most powerful feature of Code Composer is the debugging function, which allows for very accurate step through of the code and a real time view of the variables as you debug. This software tool will greatly enhance the coding process.

Within Code Composer Studio (figure 24) there is a graphical user interface composer that can be used to program the user interface for the UCF. This simple to use add-on to code composer will allow the group to design and develop the GUI, so that it can be implemented in the prototype as well as the final product. Within this program the group will be able to develop the Graphics of the GUI. Then, GUI Composer will give the programmer the source code in C language, which can be uploaded into the development board.

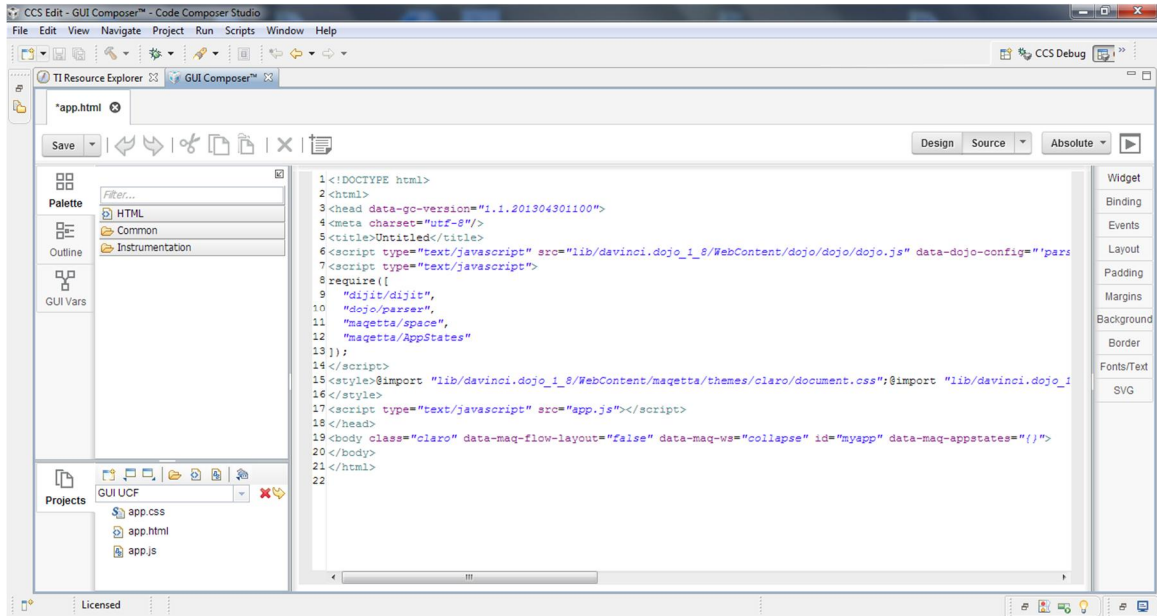


Figure 24 – Code Composer Studio IDE (with permission from Texas Instruments)

8.2 FINAL DESIGN

This section will cover the final design implementations of the Universal Circuit Fabricator. First, it will describe the differences between the hardware and software development between the prototyped and final design and will also include the following: PCB Layout Design, as well as PCB vendor and assembly.

8.2.1 Hardware Final Design Differences

Regarding the development of the UCF, moving on from the prototyped design, the UCF final design went through some changes regarding what components will be involved in the final iteration and its PCB. Due to lack of memory and difficulty of integration as well as a change in scope to the project, some components, like the SD Card reader and the LCD touch screen input was left behind. However, the final design still has the four main subsystems discussed above – the motor control system, the USB-to-Serial chip, the InkShield and inkjet control system, and finally, the ATmega328P microcontroller itself.

A new motor control system was also selected instead of the BOOST DRV8711 because of this change. The new Motor Control system features a PWM Driver, and two TB6612FNG DC motor drivers. The PWM Driver allows us to only utilize two analog pins from the microprocessor. The advantage of utilizing the TB6612FNG motor driver is that it creates lower voltage drops across the motors to increase torque and accuracy. The motor driver chip contains 2 H-Bridges, which allows the UCF to utilize bidirectional motors. The H-Bridge motor drivers will control two NEMA 17 stepper motors for the X and Y axes. The use of only two analog pins makes the use of this PWM driver essential in the design of the Universal Circuit Fabricator.



Figure 25 – TB6612FNG Motor driver changed in final design

8.2.2 Software Final Design Differences

Within the UCF's software, changes also had to be made when the hardware was changed. Most importantly, the serial communication between the host PC and the UCF is now only through USB-to-Serial communication through the ATmega16U2 chip found in the prototype development board.

The software of the Universal Circuit Fabricator interacts with the user via serial communication over USB with an external computer. The software is designed to allow the user to create, on an external computer, a text file (.txt) that contains G-Code commands which describe the design to be printed. The user then opens GcodeSender, a java based serial communication graphical user interface, on an external computer to connect to the USB COM port that the Universal Circuit Fabricator is interfaced with. Once the UCF has been successfully connected, the main menu is displayed in the graphical user interface of GCodeSender. This allows the user to type and send individual lines of GCode at a time. GCodeSender also allows the user to select a .txt file from the hard drive of the computer. Once selected, the text file is sent to the UCF one line at a time until every command has been performed.

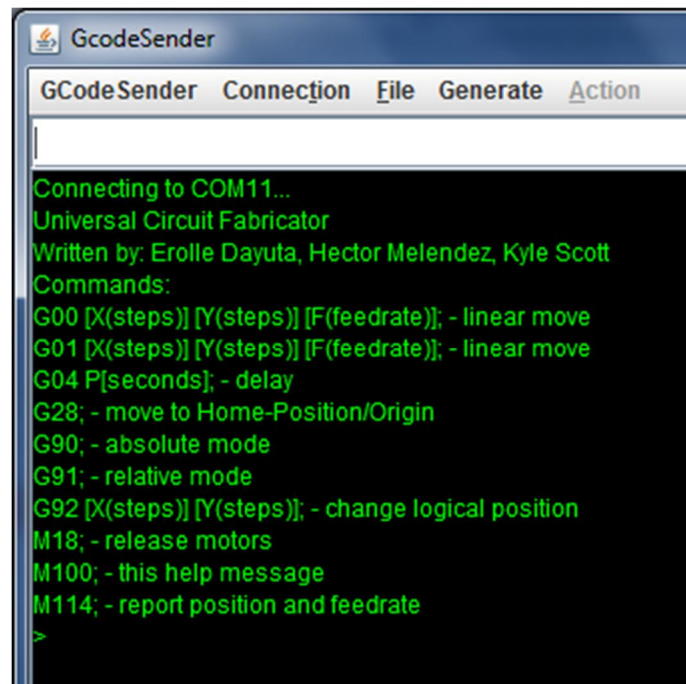


Figure 26 – Gcode Sender GUI of UCF

8.2.3 PCB Layout Design

CADSOFT Eagle was utilized in order to create the custom printed circuit board (PCB) design for the Universal Circuit Fabricator. Figure 27 shows the schematic layout in Eagle. The PCB houses all the subsystems of the UCF, as well as the ATmega328P microcontroller. The PCB design contains a USB input for both power and communication between the external PC and the UCF, powered by

the ATMEGA16U2 chip which acts as a serial to USB converter. It also contains two TB6612FNG motor control drivers which power the two stepper motors. Finally, it includes a series of headers from the I/O pins of the ATmega328P that interfaces directly with the InkShield in order to provide seamless integration. The primary design considerations were simplicity to control costs, as well as functionality within the original design. For those reasons, it was decided that the primary power supply of the system would remain off the board in order to reduce troubleshooting. The PCB manufacturer chosen for the UCF was OSHPark as it provided great service for a reasonable price, as well as providing three copies of the printed circuit board in case an error in assembly occurred. Figure 28 shows the PCBs that were shipped from OSHPark.

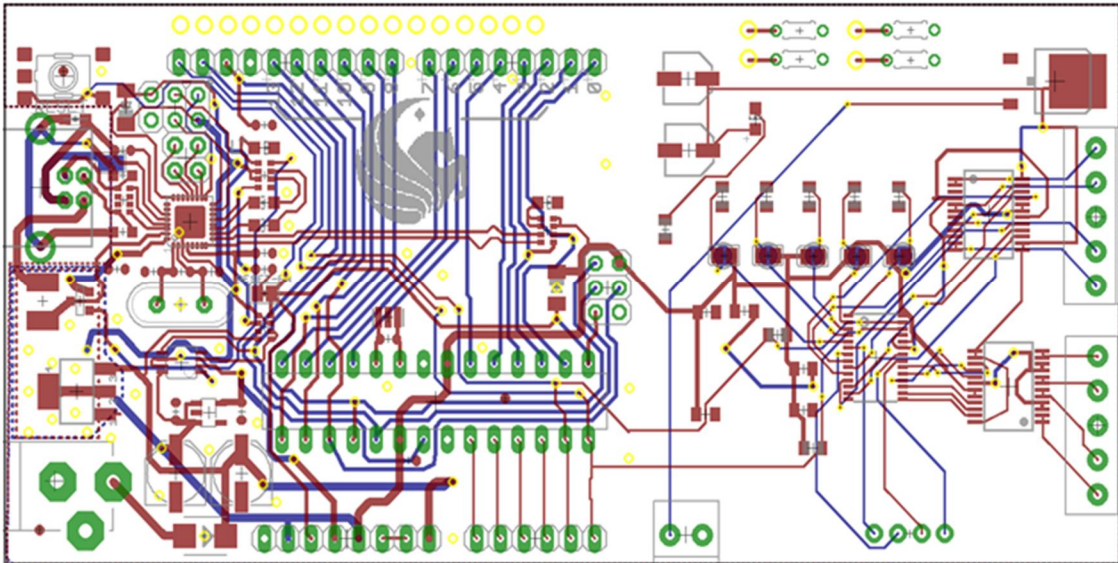


Figure 27 – PCB Layout Design in CADSOFT Eagle



Figure 28 – UCF Custom Printed Circuit Board from OSHPark

Table 20 contains the names of the theoretical connections that the PCB will contain, along with a small description of each connection.

Table 20 - Description of PCB Connections

Name	Description
XMO1, YMO1, XMO2, YMO2	Connectors for bipolar stepper drivers
JUP2, JUP3	Micro-stepping mode jumpers.
XOUT, YOUT	Breakout headers for stepper slots 1. GND 2. DIR 3. STEP 4. ENABLE
MPOWER	Connection to power the two stepper motors
5V	5V output
12V	12V output
RSTEXT	Header to connect an external reset button
JP5V	Power source selection. This determines how the PCB will be powered.
AUX1	Analog/Serial output
USB	Serial bus connection from host PC
DC21MMX	Barrel Jack 5V Input
PWRB	Header to enable and disable the power supply

8.2.4 PCB Parts List

The following Table is a list of all the major parts that will be used in the PCB of the Universal Circuit Fabricator. It will also contain a small description of the component, the manufacturer and the part number. The actual PCB was manufactured through OSHpark – chosen because of its reliability, reasonable price, and for providing three copies of the PCB in case mistakes or errors were made in assembling. Table 21 lists the parts that were used in assembling the custom PCB.

Table 21 – PCB Components

PCB Item	Manufacturer	Description	Part Number	Qty
Microcontroller	Arduino	Arduino UNO Microprocessor	Atmel ATmega328P	1
Motor Control	Toshiba	Dual DC motor driver	TB6612FNG	1
PWM Driver Chip	NXP	Handles motor and speed controls over I2C	PCA9685	
Serial-to-USB Chip	Atmel	Interface between host PC and microcontroller	ATmega16U2	1
USB Connector	Digi-Key	Micro-USB connector	WM1398TR-ND	1
Power Connector	sparkfun	DC barrel 12V power connector	PRT-00119 ROHS	1
Passive Components	Varies	Passive components (resistors, capacitors, etc.) used in the circuit schematic	N/A	Varies

8.2.5 Final Design Housing and Frame

Finally, all of the various hardware and software components were designed and attached to a printing frame, a series of pulleys and ribbons that allowed the stepper motors to move the ink nozzle print head. The actual inkjet nozzle was attached to the middle of the frame in order to move freely when the stepper motors moved. A wooden housing was also designed and built by the group in order to allow the UCF to move and be transported easily. The PCB itself was attached to the side of the housing, and all connections were securely made in order to prevent unnecessary and unrecognized movement.

Hardware switches were even mounted to the frame in order to create limits for the stepper motors and allow the motors to transmit their position to the microprocessor at all times. Figure 29 shows the final design of the UCF, including the discussed frame, as well as the housing that covered the whole project. On the left of the figure, the UCF power supply is seen, as well as the custom Printed Circuit Board.

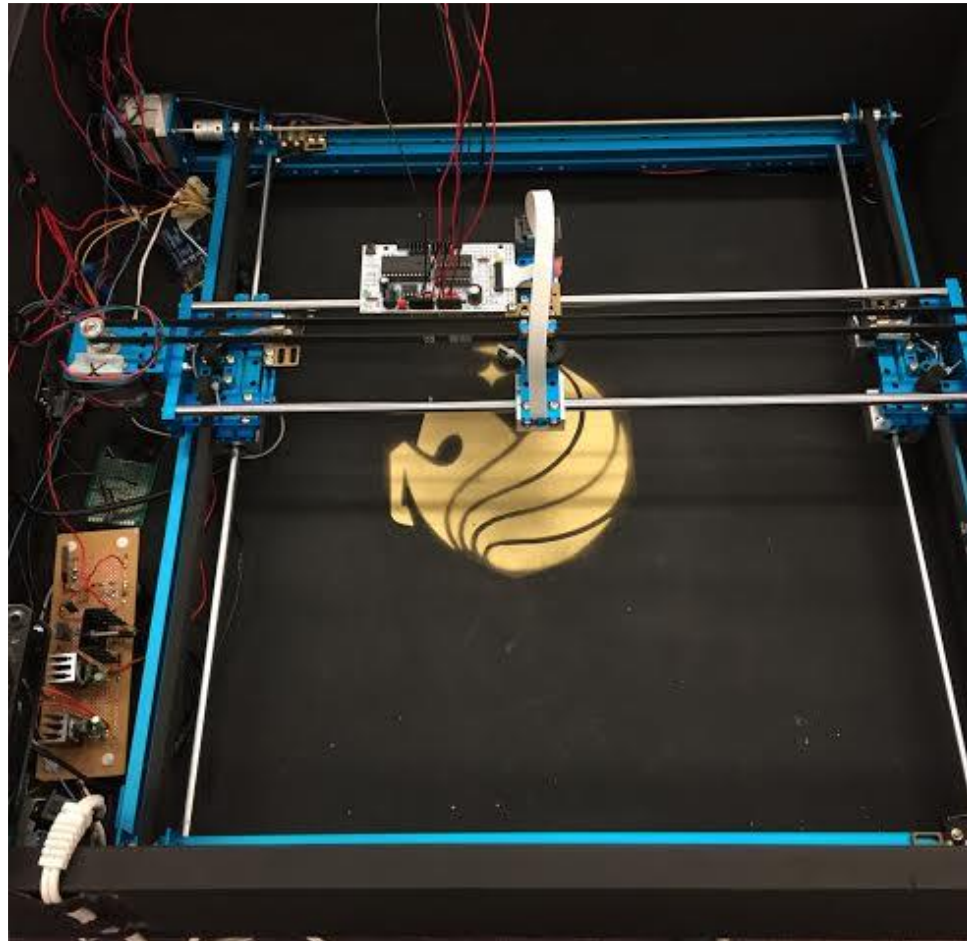


Figure 29 – Universal Circuit Fabricator Frame and Final Design

9 Design Constraints and Standards

9.1 REALISTIC DESIGN CONSTRAINTS

The following sections cover the realistic design constraints that the Universal Circuit Fabricator includes.

9.1.1 Economic Factors

The Universal Circuit Fabricator was assigned a budget based on initial research and part selection. Like every new project, it needs to have a budget for spending and keeping the cost the product within a reasonable range. But the development and construction of a project does not always goes as planned. The UCF had to change the architecture of the microprocessor, in order to incorporate a major portion of the design, the Inkshield. Also, the cost of chemicals needed to create the conductive ink was not properly addressed. Because, several of the chemicals needed could only be purchased in bulk amounts, which was not considered in the initial budget request.

9.1.2 Manufacturability

The Universal Circuit Fabricator was designed to create circuits on a non-conductive surface. This allows it to be used for the manufacturing of printed circuit boards, within the confines of the user's home. The UCF can also be used to manufacture printed circuit boards commercially, with further development. The UCF is meant to be a proof of concept. That the user can print their own circuit schematics on a piece of glass, rather than have to build their circuit on a breadboard and then have it manufactured by a company the makes PCBs.

9.1.3 Health and Safety

Engineers need to have guidelines to ensure the safety of use of their projects. The Universal Circuit Fabricator utilizes a mixture of chemicals that can be harmful to the user, if ingested. Also, the creation of the conductive ink is a delicate process that requires the use of protection while producing a batch of ink. Therefore, there is a warning that the user should not come into contact with the conductive ink while it still is in liquid form. After the annealing process has taken place the ink is safe to touch, but it is not safe to be ingested.

9.1.4 Reliability

The Universal Circuit Fabricator was designed to be settled on a stationary location, while still being small enough to be easily relocated. The UCF needs to have access to a wall outlet in order to function, with further development it could work on batteries. The circuits that the UCF creates, are able to conduct electricity right after being annealed. But, it is suggested that the user allows several hours of curing to fully get the maximum conductivity. As for the durability of the ink, it is recommended that the user does not touch the annealed traces,

because this will cause the traces to be discontinuous. If the user utilizes the circuit correctly, the printed circuit should last for a long time.

9.1.5 Environmental Impact

The Universal Circuit Fabricator utilizes glass as its printing substrate, which has some advantages. The first is that glass can be easily purchased anywhere, unlike the typical substrate used in the industry, FR-4. Also, once the conductive ink had annealed, the ink can be removed from the glass by simply washing it with water and a paper towel. This makes the glass re-usable for printing more circuits. Therefore it is more environmentally friendly than other substrates that need to be disposed of.

9.1.6 Ethical

The purpose of the Universal Circuit Fabricator is to create your own printed circuit boards, but that comes with certain limitations. The user can print any circuit schematic that they desire, but it must not be any copyrighted design belonging to any separate entity. The UCF is intended to be used for the purpose of prototyping and development of innovative circuit boards. Not for the use of printing copyrighted material and to be sold for personal gain. There is an ethical and legal boundary that the user needs to be aware of before utilizing the UCF. Meaning, that there is no limit to what the user can do with the Universal Circuit Fabricator.

9.2 STANDARDS

There were several standards that have been developed by engineers in industry that were applicable when designing the Universal Circuit Fabricator Prototype. Industry Standards simplify the design process greatly by verifying that the parts that were selected have been designed to work when interfaced with a wide range of other products. Table 22 highlights a few standards documents that detail the uniform functionality of the parts and concepts that were used in the UCF Prototype.

Table 22 – Standards Observed in UCF Prototype Design

Standard	Title	Document Number
Serial Communication	IEEE Standard for a High-Performance Serial Bus	IEEE 1394-2008
USB Connection	Universal serial bus interfaces for data and power - Part 1: Universal serial bus specification, revision 2.0	IEC 62680-1 Ed. 1.0 b:2013
Circuit Board Design	Device embedded substrate - Part 2-3: Guidelines - Design guide	IEC/TS 62878-2-3 Ed. 1.0 b:2015
NEMA – Stepper Motor Selection	NEMA Standards Publication Motors and Generators	BSR NEMA MG-1-201x
G-Code	Numerical control of machines - Data format for positioning, line motion and contouring control equipment (FOREIGN STANDARD)	AS 1114.1-1985

10 User Manual

The user manual section details the steps that must be taken for successful operation of the Universal Circuit Fabricator prototype. The hardware components must be successfully integrated with the custom made PCB. Once the hardware has been integrated, the software that supports the UCF prototype must be installed on the user's computer and the input file must be generated and sent to the PCB to initiate printing.

10.1 HARDWARE SETUP

There are several hardware subsystems involved in the Universal Circuit Fabricator prototype design. In order to successfully operate the prototype, all hardware subsystems must be correctly interfaced together.

10.1.1 USB Serial Interface

The USB Serial Interface allows the host computer to communicate with the Universal Circuit Fabricators' custom PCB. For this step, a USB cable with a Type A connector on one end and a Type B connector on the other end is required. The user must connect the Type A USB to the host computer and the Type B USB to the custom PCB located on the Universal Circuit Fabricator.

10.1.2 Stepper Motor Interface

The Stepper Motor Interface comprises of the screw terminals that are mounted on the custom PCB for the Universal Circuit Fabricator. There are eight of these screw terminals, four for each stepper motor. Each stepper motor contains two independent internal coils which influence the precision of the motors; each coil has two wires associated with it. The red and blue wires are associated with one coil while the green and black wires are associated with the second coil. It is imperative that the wires associated with the coil are kept together when interfacing with the PCB, otherwise the coils will work against each other, preventing movement of the motors. Refer to figure 30 for the correct orientation of the wiring for the Stepper Motor Interface.

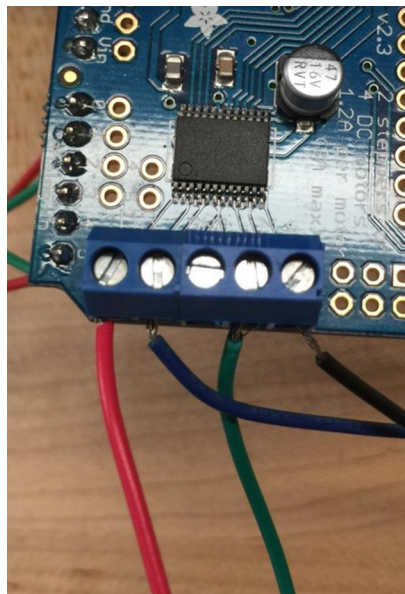


Figure 30 – Wire Configuration for Stepper Motor

10.1.3 InkShield Interface

The InkShield board is integral in the Universal Circuit Fabricator Prototype as it is responsible for controlling the flow of ink out of the inkjet cartridge. In the final design of the prototype, the InkShield is mounted on top of the print carriage and translates about the print bed as a design is being printed. In order to successfully control the flow of ink, the InkShield must be connected to the UCF's

custom PCB via Digital Pin D2 as well as Analog Pins A0, A1, A2, and A3. Additionally, the InkShield is powered via the Vin pin and grounded through the Grd pin of the custom PCB. In total, seven wires must run from the custom PCB to the InkShield. It is required that these wires be long enough so that the carriage where the InkShield is mounted can move around the entire print bed without disconnecting the wires.

10.1.3.1 Filling Inkjet Cartridge with Conductive Ink

The HP6602 inkjet print cartridge comes pre-filled with black ink. This is useful for testing the prototype, but must be replaced when printing conductive ink is desired. Gloves are required for the preparation of the inkjet cartridge. Using a hand saw or a dremel cutting tool, the top of the HP6602 must be cut off. Once opened, the sponge containing the black ink must be removed and washed with water to remove all of the black ink. Once the inside of the cartridge and the sponge are cleaned, the inkjet cartridge is put back together and the top is once again made airtight by using glue or tape. The cartridge is then filled with conductive ink by using a needle syringe and the small air inlet hole that is located on the top of the HP6602.

10.1.4 Hardware Switch Interface

The hardware limiting switches are required for setting the origin of the Universal Circuit Fabricator coordinate system as well as signaling when the carriage has reached the end of the frame, preventing damage to the frame or the motors. Each hardware switch has three terminals. The Vcc terminal for each switch must be wires together in parallel and requires 5v DC from the power supply. The ground terminal for each switch must independently pass through a pull down resistor of 1k Ω and then are tied to the common ground of the power supply. The third terminal of the hardware switch must be connected to the custom PCB to send 5v DC signal to the software when the switch is pressed. Based on the software configuration, the switches must be wired according to table 23.

Table 23 – Wiring configuration for hardware switches

Hardware Switch	Digital Pin on PCB
Limit Switch End for X axis	D3
Limit Switch Home for X axis	D7
Limit Switch End for Y axis	D6
Limit Switch Home for Y axis	D4

10.2 SOFTWARE SETUP

While the software for the Universal Circuit Fabricator is stored on the microprocessor on the custom PCB, there are certain steps that must be taken to interface with the prototype and print a custom circuit schematic.

10.2.1 Circuit Trace Generation

The desired circuit trace to be printed is generated using a drawing program such as Microsoft Paint on the user's host computer. The schematic must be saved as a bitmap picture file (.bmp). Figure 31 shows an example of a circuit trace bitmap that was generated in Microsoft Paint.

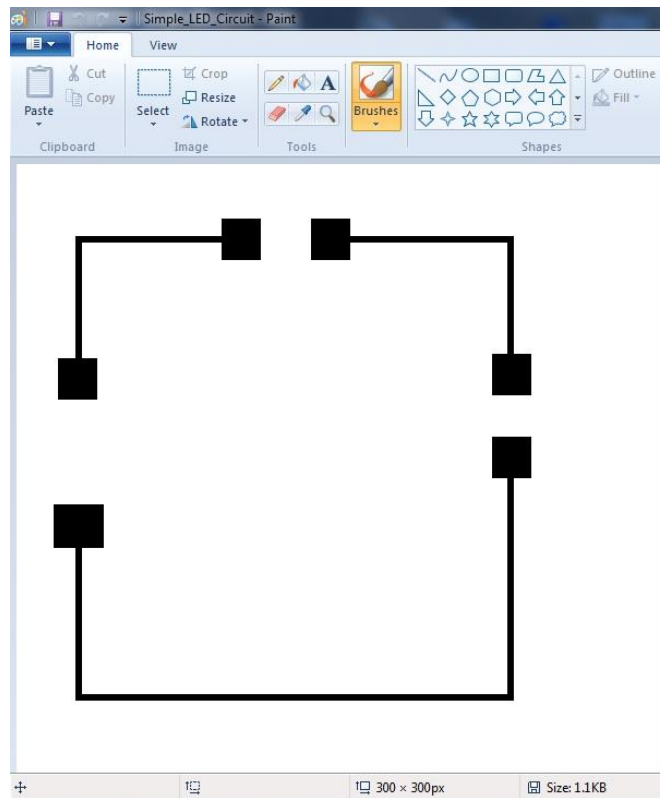


Figure 31 – Example circuit schematic

10.2.2 Conversion to G-Code

The user must download an open source vector based graphics editor called Inkscape. The program can be downloaded and installed for free. Additionally, an open source plug in for Inkscape called Gcodetools must be downloaded and installed. Once installed successfully, the schematic bitmap can be imported into Inkscape and the path can be traced and converted into GCode. Figure 32 shows the path that was traced around the example circuit schematic in Inkscape. Figure 33 shows the G-Code that was generated as a result of using the Gcodetools plugin.

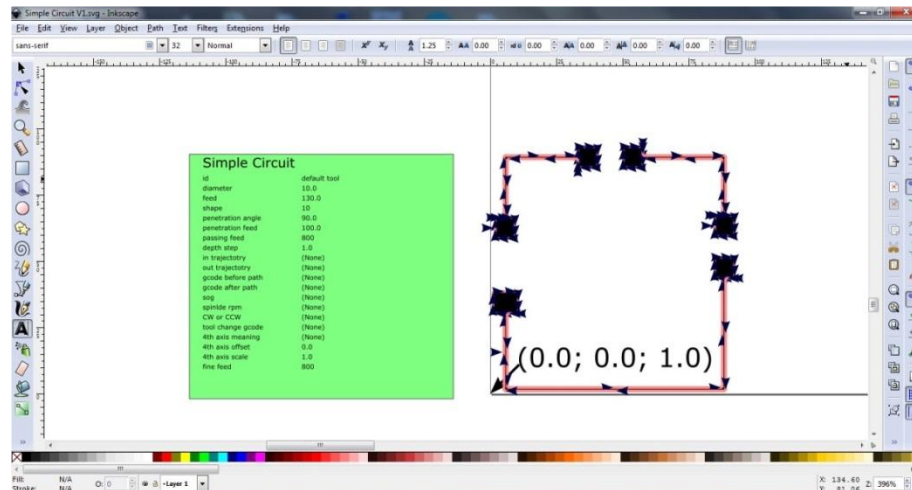


Figure 32 – Inkscape converting bitmap to G-Code

```

circuit demo.txt - Notepad
File Edit Format View Help
G28;
G00 X1.965476 Y4.720704 F130.0;
G01 X1.965476 Y8.836436 F130.0;
G01 X1.377513 Y8.836436;
G01 X0.789550 Y8.836436;
G01 X0.789550 Y10.012371;
G01 X0.789550 Y11.188306;
G01 X2.141865 Y11.188306;
G01 X3.494180 Y11.188306;
G01 X3.494180 Y10.012371;
G01 X3.494180 Y8.836436;
G01 X2.906217 Y8.836436;
G01 X2.318254 Y8.836436;
G01 X2.318254 Y4.897093;

```

Figure 33 – G-Code generated by Inkscape

10.2.3 G-Code Command Transmission

Once the circuit schematic has been created and converted to G-Code commands, it is ready to be sent to the Universal Circuit Fabricator to be printed. This process utilizes the USB Serial interface on the host computer and custom PCB. In order to facilitate communication and transmission of the G-Code, the GcodeSender program must be downloaded. GcodeSender is an open source Java based GRBL compatible cross platform G-Code sender. It allows the user to view the menu for the UCF, send individual G-Code commands, send text files containing multiple G-Code commands, and see feedback on the position of the print head. The graphical user interface for GcodeSender is shown in figure 26. Once the hardware connection has been made, the corresponding COM port is selected and the UCF menu is displayed. The file menu allows the user to select the text file containing G-Code commands that has been generated in previous steps. The action menu contains the run option which will begin the printing process.

11 Test Plan

Testing each component of the Universal Circuit Fabricator is crucial to ensure that the subsystems function individually so that they will function successfully when integrated as a system. Rigorous testing of the systems at a component level greatly reduces debugging effort when the subsystems are integrated into the overall design. The functioning prototype must also be tested against the design specifications to ensure that the Universal Circuit Fabricator delivers the promised functionality.

11.1 HARDWARE TESTING

The subsystem hardware will be tested against the design requirements and specifications to verify that the prototype functions as designed. The Data Input and Processing, Motor Control, Inkjet Cartridge Control System, Annealing, and Graphical User Interface hardware subsystems must pass the following test cases in order to function desirably as a system.

11.1.1 Data Input and Processing Hardware

The Data Input and Processing subsystem hardware must be able to process the user's design either via SD Card or the Capacitive Touch sensor. To ensure hardware functionality, the SD Card interface with the microcontroller must be verified functionally. The Capacitive Touch Sensor hardware also must be tested to ensure it can interface the user with the embedded processor. The following test cases will verify that the hardware functions as needed.

11.1.1.1 Data Input and Processing Hardware Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 24 – Data Input and Processing Hardware Test Case 1

Test Objective	SD Card Breakout Board Integration with Prototype Microcontroller
Test Description	Verification that the SD Card slot hardware has been successfully integrated with the hardware of the Arduino UNO microcontroller
Test Conditions	The SD Card slot is wired to one of the Arduino prototyping microcontroller's I/O pins. Using Code Composer Studio, a file is written to the SD Card. The card is then removed and

	tested on a separate computer to ensure that the file was successfully written. The SD Card is then reinserted into the SD Card slot connected to the Arduino and the file is read using Code Composer Studio.
Expected Results	The SD Card slot breakout is able to write and read to/from the SD Card. The file that was written on the SD Card is able to be viewed from an external computer. The SD Card will be removed and reinserted into the breakout slot and the file will be read again successfully.

Table 25 – Data Input and Processing Hardware Test Case 2

Test Objective	Capacitive Touch Sensor Integration with Prototype Microcontroller
Test Description	Verification that the Capacitive Touch Sensor hardware has been successfully integrated with the hardware of the Arduino UNO prototype microcontroller.
Test Conditions	The Capacitive Touch Sensor is wired to one of the Arduino prototyping microcontroller's I/O pins. Using Code Composer Studio, the input value of the sensor is observed while there is nothing touching the sensor. After establishing a baseline value, a member of the group will interact with the sensor by touching it.
Expected Results	The baseline measurement of the Capacitive Touch Sensor will be less than the measurement observed when the group member interacts with it.

11.1.2 Motor Control Hardware

The Motor Control subsystem hardware must be able to move the ink jet print head in both the X and Y axis. The motors that are used in the motor control subsystem must be able to be integrated with the hardware of the microcontroller. They must have enough precision to move the print head one millimeter at a time in order to meet the conductive trace thickness specification. The motors must also have the range to move the print head the full five inches required to cover the print bed area. The following test cases will verify that the hardware functions as needed.

11.1.2.1 Motor Control Hardware Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 26 – Motor Control Hardware Test Case 1

Test Objective	Stepper Motor Movement
Test Description	Verification that the Stepper Motor hardware has been successfully integrated with the hardware of the Arduino prototype microcontroller.
Test Conditions	The Motor Controller breakout board is wired to one of the Arduino UNO prototyping microcontroller's I/O pins. Two Stepper Motors are then wired to the Motor Controller breakout board. Using Code Composer Studio, a program is written to step the motors in the forward and reverse directions
Expected Results	Both the X-axis and Y-axis motors can function in the forward and reverse directions.

Table 27 – Motor Control Hardware Test Case 2

Test Objective	Stepper Motor Precision
Test Description	Verification that the Stepper Motor hardware provides the precision necessary for printing thin traces.
Test Conditions	Following the Motor Movement test case, the Stepper Motors are verified to move in the forward and reverse directions. The Motors are now connected to the belts that will translate the print head across the print bed area. Using Code Composer Studio, a program is written to step the motors one step at a time with a delay.
Expected Results	The Stepper Motor moves the translation belt with the precision of 1 mm/step corresponding to the conductive trace line thickness specification.

Table 28 – Motor Control Hardware Test Case 3

Test Objective	Stepper Motor Range
Test Description	Verification that the Stepper Motor hardware has the range to cover the print bed area.
Test Conditions	Following the Stepper Motor Precision test case, the Motor Control Subsystem has been verified to function with acceptable precision. The Motors are still connected to the translation belts. A program is written in Code Composer Studio that steps the motor in the forward direction until one of the buttons on the prototype board is pressed. When the button is pressed, the program will step the motor in the reverse direction until the button is pressed again.
Expected Results	The Stepper Motor is able to translate the belt a minimum of 5 inches corresponding to the print bed area specification.

11.1.3 Inkjet Cartridge Control Hardware

The Inkjet Cartridge subsystem hardware must be able to print conductive ink using a HP C6602 ink cartridge. In order to accomplish this, the ink shield breakout board kit must be assembled and verified as functioning as designed. The functioning ink shield hardware must then be successfully integrated with the hardware of the microcontroller to complete the Inkjet Cartridge Control subsystem. The following test cases will verify that the hardware functions as needed.

11.1.3.1 Inkjet Cartridge Control Hardware Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 29 – Inkjet Cartridge Control Hardware Test Case 1

Test Objective	Assemble functioning InkShield breakout board
Test Description	Verification that the Inkjet Cartridge Control hardware has been successfully assembled.
Test Conditions	The InkShield Breakout board is shipped as a PCB with several discreet components that must be soldered to the board. The board is assembled according the instructions provided and is powered using a Arduino UNO Vout pin. The instructions provided with the Inkshield point out three junctions on the PCB to check for reference voltages.
Expected Results	<ol style="list-style-type: none"> 1. The “+5” Volt junction displays 5V on a multimeter in reference to ground 2. The “+12” Volt junction displays 12V on a multimeter in reference to ground 3. The 20 Volt junction for powering the print head displays 20V on a multimeter in reference to ground

Table 30 – Inkjet Cartridge Control Hardware Test Case 2

Test Objective	InkShield breakout board integration with Prototype Microcontroller.
Test Description	Verification that the Inkjet Cartridge Control hardware has been successfully integrated with the hardware of the Arduino UNO prototype microcontroller.
Test Conditions	The assembled and functioning InkShield breakout board is wired to one of the Arduino Prototype Microcontroller's I/O pins as well as the Vout pin. A program is written using Code Composer Studio that utilizes the open source Arduino code provided with the InkShield. The code provided with the InkShield instructs the inkjet print head to spray all nozzles as fast as possible.
Expected Results	The effect of this test will be a constant stream of ink flowing from the print head. A group member will move the cartridge/print head assembly around a piece of blank paper and observe a continuous line of ink following the pattern of movement.

11.1.4 Annealing Hardware

The Annealing subsystem hardware must be able to heat the print bed area to the temperature required for fusing the conductive ink into a continuous trace. The hardware for reading and maintaining the annealing temperature in the print bed must be tested. The following test cases will verify that the hardware functions as needed.

11.1.4.1 Annealing Hardware Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 31 – Annealing Hardware Test Case 1

Test Objective	Thermocouple integration with Prototype Microcontroller
Test Description	Verification that the thermocouple that will be used to regulate the annealing subsystem reads temperature accurately and communicates with the microcontroller.
Test Conditions	The Thermocouple is wired to one of the Arduino UNO Prototype Microcontroller's I/O pins. A program is written in Code Composer Studio that sets the subject I/O pin as an input and displays the current reading on a terminal session with a 1 second interval.
Expected Results	When the program is run, the terminal screen will display the room's temperature and refresh every second. A group member will hold the thermocouple in their fingers to raise the temperature, and the change in temperature is observed in the terminal.

Table 32 – Annealing Hardware Test Case 2

Test Objective	Hot Plate reaches annealing temperature.
Test Description	Verification that the hot plate that will be used for the annealing subsystem can reach the required temperature
Test Conditions	The hot plate is powered using power from a standard outlet and set to reach its maximum temperature. The thermocouple that was integrated with the Microcontroller is used to measure the temperature of the hot plate surface.
Expected Results	The hot plate will reach and maintain a temperature that exceeds 200F =~ 93C

11.1.5 Graphical User Interface Hardware

The Graphical User Interface subsystem hardware must be able to allow the user to interact with the microcontroller to operate the Universal Circuit Fabricator. The LCD display must be tested to verify that it can be integrated with the microcontroller. The following test cases will verify that the hardware functions as needed.

11.1.5.1 Graphical User Interface Hardware Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 33 – Graphical User Interface Hardware Test Case 1

Test Objective	LCD Display Integration with Prototype Microcontroller
Test Description	Verify functionality of LCD screen hardware
Test Conditions	The LCD Display is wired to one of the Arduino UNO Prototype Microcontroller's I/O pins. Power is supplied to the LCD screen so as to verify the hardware can be powered using the embedded microcontroller.
Expected Results	When power is supplied to the LCD screen, it will light up with test screen verifying that it can be powered using the microcontroller.

11.1.6 Final Hardware System Evaluation

Upon passing all hardware test cases, the functionality of the individual hardware subsystems has been verified. The next phase of testing begins by testing the software that is developed to integrate and control the functioning hardware.

11.2 SOFTWARE TESTING

There are several software systems involved in controlling the Universal Circuit Fabricator that must interact with each other to control the hardware systems in a desired fashion. After passing the hardware test cases that verified that the hardware functions at a basic level, the software testing verifies the programs that have been developed to control the hardware successfully control functioning sub-systems designed specifically for the UCF.

11.2.1 Data Input and Processing Software

The Data Input and Processing subsystem software must be able to process the user's design that has been provided via either SD Card or the Capacitive Touch sensor. The Data Input and Processing software must be able to read an input file from the SD Card as well as register input from the capacitive touch sensor. The software's ability to input the correct file to the microcontroller's internal memory will be tested as well as the software's ability to output the file from the microcontroller's internal memory to the SD Card. The following test cases will verify that the software functions as needed.

11.2.1.1 Data Input and Processing Software Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 34 – Data Input and Processing Software Test Case 1

Test Objective	SD Card transmits desired file type to Microcontroller's internal memory
Test Description	Verify that the functioning SD Card slot / microcontroller interface can successfully transmit the design file type from SD Card to microcontroller memory.
Test Conditions	Depending on the file type that is selected for the UCF (jpg, bmp, png, gbr), A design file is saved on a SD Card using an external computer. The SD Card is inserted into the embedded microcontroller. Using the Code Composer Studio program that was developed to read the input file, the design is verified as a valid file that is ready to be printed.
Expected Results	When the file is verified as valid by the software, a LED will light on the Arduino UNO prototyping microcontroller.

Table 35 – Data Input and Processing Software Test Case 2

Test Objective	Capacitive Touch Sensor transmits designs to Microcontroller's internal memory
Test Description	Verify that the functioning capacitive touch sensor / microcontroller interface can successfully transmit a design that was created using the capacitive touch sensor to the internal memory of the embedded microcontroller.
Test Conditions	Using the Code Composer Studio program that was developed to read the user's input on the capacitive touch sensor, a file of the desired file type is created and saved into the internal memory of the Arduino UNO prototyping microcontroller.
Expected Results	A group member will draw lines on the capacitive touch sensor and push a button on the Arduino UNO that is assigned by the program to save the design. The internal memory of the microcontroller is then observed using Code Composer Studio to verify that a design has been saved into memory.

Table 36 – Data Input and Processing Software Test Case 3

Test Objective	Microcontroller transmits desired file type to SD Card
Test Description	Verify that the functioning microcontroller / SD Card slot interface can successfully transmit a design that was created using the capacitive touch sensor to the SD Card memory.
Test Conditions	Depending on the file type that is selected for the UCF (jpg, bmp, png, gbr), a design file is saved directly in the microcontroller's internal memory from an external computer. A blank SD Card is inserted in the embedded microcontroller. Using the Code Composer Studio program that was developed to write the output file, the file is saved onto the SD card.
Expected Results	Once the file has been written on the SD Card, it can be removed and read by an external computer to verify that the file was successfully transferred.

11.2.2 Motor Control Software

The Motor Control subsystem software must be able to move both the X-axis and Y-axis motors according to the parameters set in the design file. The software's ability to move both the X-axis and Y-axis motors in both the forward and backward direction will be tested. In addition, the software will be tested in its ability to move the inkjet print head in a pattern that corresponds with an input design file. The following test cases will verify that the software functions as needed.

11.2.2.1 Motor Control Software Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 37 – Motor Control Software Test Case 1

Test Objective	Software controls movement of motors.
Test Description	The microcontroller must move the stepper motor for both the X-axis and Y-axis in both the forward and backward direction.
Test Conditions	Using the Code Composer Studio program that was developed to control the movement of the stepper motors, the software must enable the motor, step the motor in the forward direction, and step the motor in the backward direction.
Expected Results	The software is able to move the motors forwards and backwards in the X and Y directions.

Table 38 – Motor Control Software Test Case 2

Test Objective	Software controls movement of the print head in a designed pattern.
Test Description	The microcontroller software must control the movement of the X and Y axis stepper motors corresponding to a design that has been saved in the internal memory of the Arduino UNO processor.
Test Conditions	Using the Code Composer Studio program that was developed to control the movement of the motors, a design file will be saved into the internal memory of the Arduino embedded microprocessor. The processor must control the output of the motor control subsystem and move the motors and print head in the pattern of the input design. The design file will include moving the print head around the outer perimeter of the 5 by 5 inch print bed.
Expected Results	The microcontroller controls the movement of the motors to move the print head around the outer perimeter of the print bed.

11.2.3 Inkjet Cartridge Control System Software

The Inkjet Cartridge subsystem software must be able to control the firing of the ink jet print jet nozzles. The software will be tested in its ability to process the input file to determine when the HP C6602 ink cartridge should deposit conductive ink. The following test cases will verify that the software functions as needed.

11.2.3.1 Inkjet Cartridge Control System Software Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 39 – Inkjet Cartridge Control System Software Test Case 1

Test Objective	Verify that the software successfully integrates the ink shield breakout board with the Arduino UNO embedded microprocessor.
Test Description	The microcontroller software must control the spray of the ink jet print head via the ink shield breakout board corresponding to a design that has been saved in the internal memory of the Arduino UNO processor.
Test Conditions	Using the Code Composer Studio program that was developed to control the firing of the ink jet nozzles, a design file will be saved into the internal memory of the Arduino UNO embedded microprocessor. The processor must control the flow of conductive ink out of the print head. The design file will fire the ink jet nozzles on and off with a delay of 0.5 seconds to illustrate the software's ability to start and stop depositing conductive ink.
Expected Results	The software will fire the ink jet nozzles with an interval of half a second. A dashed line of ink can be drawn on a piece of paper by slowly moving the print head across the page.

11.2.4 Annealing System Software

The Annealing subsystem software must control when the heat bed is turned on. It must regulate a set annealing temperature for a programmed period of time. The following test cases will verify that the software functions as needed.

11.2.4.1 Annealing System Software Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 40 – Annealing System Software Test Case 1

Test Objective	Hot Plate integration with Prototype Microcontroller
Test Description	Verification that the hot plate that will be used for the annealing system can be controlled by the prototype microcontroller and regulated based on feedback from the thermocouple
Test Conditions	The hot plate is wired to one of the Arduino UNO Prototype Microcontroller's I/O pins. A program written in Code Composer Studio turns on the hot plate and uses the thermocouple as a feedback loop to regulate the power supplied to the hot plate and maintain a constant temperature. A timer is integrated into the program that will turn off the hot plate after the annealing process has been completed.
Expected Results	The annealing system maintains a constant temperature of 200F = ~ 93C for the length of the programmed timer (~1-5 minutes based on ink testing).

11.2.5 Graphical User Interface Software

The Graphical User Interface subsystem software must be able to display the menu that has been designed in the software. The LCD display must be powered and controlled using the program written in the Arduino UNO embedded microprocessor. Displaying the menu and navigating menu options will be tested. The following test cases will verify that the software functions as needed.

11.2.5.1 Graphical User Interface Software Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the subsystem that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 41 – Graphical User Interface Software Test Case 1

Test Objective	Graphical User Interface is displayed on LCD
Test Description	Display the Graphical User Interface on the LCD display and navigate the menu using the capacitive touch sensor.
Test Conditions	The Arduino UNO embedded microprocessor is flashed with the software that was developed using Code Composer Studio. The LCD display is wired to an I/O port of the processor. The menu of the written software is displayed on the LCD. Using the functioning Data Input and Processing subsystem, the user must be able to navigate the Graphical User Interface menus using the capacitive touch sensor.
Expected Results	The user is able to navigate the Graphical User Interface menu options using the capacitive touch sensor. The GUI is displayed on the LCD display.

11.2.6 Final Software System Evaluation

Upon passing all software test cases, the functionality of all software subsystems is verified. The next phase of testing begins by testing the functionality of the Universal Circuit Fabricator as a working prototype.

11.3 OVERALL SYSTEM TESTING

Overall System Testing involves testing the Universal Circuit Fabricator's ability to print a circuit from start to finish. Both SD card and Touch Screen inputs will be tested to verify that all functions of the prototype operate successfully. Multiple applications of printed circuits are created to demonstrate the UCF's ability to create practical designs.

11.3.1 Overall System Test Cases

Test Cases are classified by four categories that describe the intent of the testing. The test objective describes the general objective or requirement that is being verified in the test. The test description gives detail on the functions of the system that will be tested. The test condition describes the parameters that the subsystem will encounter during testing. The expected results describe the outcome that is desired from the test which contributes to the overall functionality of the prototype.

Table 42 – Overall System Test Case 1

Test Objective	Print a 1 cm length line of capacitive ink from a SD Card Design
Test Description	Using the input of a design file on an SD Card, print a line of conductive ink onto a glass substrate, and anneal the ink to its finalized state.
Test Conditions	The prototype must function as an integrated system with the embedded microcontroller supporting the subsystems as designed.
Expected Results	A 1 cm continuous trace is printed onto the glass substrate.

Table 43 – Overall System Test Case 2

Test Objective	Print a line from a Capacitive Touch Sensor Design
Test Description	Using the input of the user drawing a line on the capacitive touch sensor, print a line of conductive ink onto a glass substrate, and anneal the ink to its finalized state.
Test Conditions	The prototype must function as an integrated system with the embedded microcontroller supporting the subsystems as designed.
Expected Results	A continuous trace is printed onto the glass substrate.

Table 44 – Overall System Test Case 3

Test Objective	Print 1 kΩ resistor
Test Description	Using the equation $R = \rho \cdot l$ where R is measure in ($\Omega \cdot \text{cm}$) and (l) is the 1 cm trace from the SD Card test case, the resistivity (ρ) of the finalized trace material is measured using a multimeter. Based on the resistivity, a length (l) of trace is calculated that will yield a value of 1 kΩ
Test Conditions	The prototype must function as an integrated system with the embedded microcontroller supporting the subsystems as designed. The trace will be designed on a computer and saved on an SD card to ensure the correct length of the trace.
Expected Results	The electrical resistance of the printed trace is measured to be 1 kΩ.

Table 45 – Overall System Test Case 4

Test Objective	Solder discrete components to printed traces
Test Description	Test the ability of the printed conductive ink to bond so a soldered connection to a discrete component.
Test Conditions	A length of conductive ink is printed with a gap in the middle. A 1 kΩ resistor is soldered to each end of the gap and the resistance is measured across the entire trace.
Expected Results	The measured resistance will be 1 kΩ.

Table 46 – Overall System Test Case 5

Test Objective	Solder discrete components to printed traces using commercial conductive paste
Test Description	In the situation where discrete components are unable to be soldered to printed ink, test the ability to solder discrete components to commercially available conductive paste.
Test Conditions	A length of conductive ink is printed with a gap in the middle. Conductive paste is applied to the ends of the gap and allowed to dry. A 1 kΩ resistor is soldered to the conductive paste on each end of the gap and the resistance is measured across the entire trace.
Expected Results	The measured resistance will be 1 kΩ.

Table 47 – Overall System Test Case 6

Test Objective	LED Circuit design
Test Description	Print a design corresponding to the circuit in the figure below.
Test Conditions	The traces will leave adequate gaps to solder a 1 k Ω resistor and a NTE30043 blue LED. Alligator clips fastened to printed conductive ink terminal pads will supply the voltage required to light the LED.
Expected Results	When 5 V is supplied across the printed terminals, the LED will light.

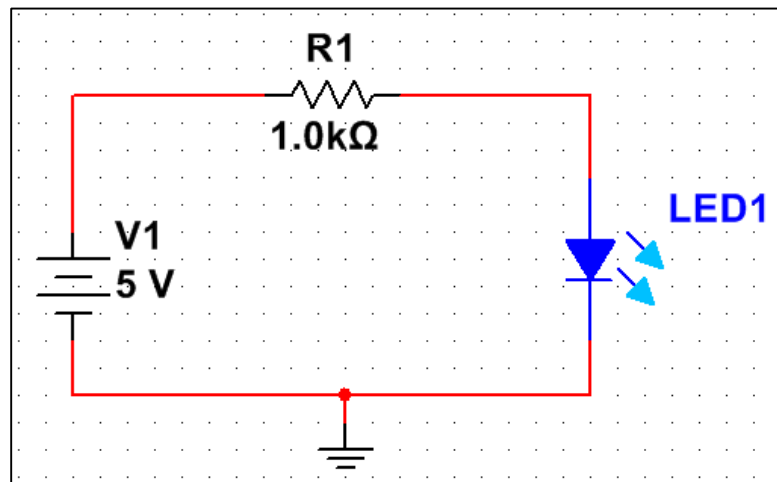


Figure 34 LED Circuit Design Schematic

11.4 OVERALL SYSTEM EVALUATION

Results of the overall system tests are compared against the design specifications to determine if the Universal Circuit Fabricator's design and build quality meets the desired requirements.

12 Budget & Financing

The following section describes the budget in regards to the bill of materials to the prototype and the final design of the UCF. The list of materials was derived from research from within the group as well as from various resources from manufacturers. The section also described the source of financing in regards to the UCF total budget.

12.1 BILL OF MATERIALS

The following list of materials was derived through extensive research, and analysis to fit the project. Some of the items will be provided by the group members, and will be marked as such in the tables below. The parts shown below are subject to change, depending on how they perform. The vendors were chosen based on the cost, reliability and reputation. While the vendors listed are the best options currently, it will depend on the availability of the parts needed.

The list below shows the hardware components required for development purposes. These parts are required to build a working prototype of the design, and then thoroughly tested.

Table 48 – Hardware Development Materials

Item	Manufacturer	Description	Part Number	Vendor	Qty	Price
TI MSP-430	Texas Instruments	Dev-Board	MSP-EXP430G2	estore.ti.com	1	Provided by group member
Arduino Uno	Arduino	Arduino Uno – ATmega328	DEV-11021	www.sparkfun.com	1	Provided by group member
InkShield Breakout Board	Nicholas C. Lewis	Inkjet Cartridge Driver	InkShield	http://nicholasclewis.com/store/	1	\$63.00
XY Plotter 2D Frame	MakeBlock	XYPlotter Robot Kit	XYPlotter Robot Kit	www.makeblock.cc	1	\$210

Development Board / Shield Headers	SparkFun	Arduino Stackable Header - 6 Pin	PRT-09280	www.sparkfun.com	5	\$2.50
Shield Headers	SparkFun	MSP430 – Stackable Headers – 8 Pin	PRT-09279	Sparkfun.com	5	\$4.00
Stepper Motor	Sparkfun	Bipolar motor, NEMA 17 form factor	ROB-10846	www.sparkfun.com	2	\$16.95/ea
LCD Display Breakout Board	43oh	2.2 inch color LCD	Color LCD Display	www.tindie.com	1	\$16.00
Capacitive Touch Shield	KENTEC ELECTRONICS	ITO Glass Capacitive Touch Sensor	KT35S430-S1	Sg.element14.com	1	\$10.82
SD/Micro SD Card Shield	43oh	SD card reader breakout board	DEV-MSP-SDCARD-	Store.43oh.com	1	\$9.99
Motor/Stepper/Servo Arduino Shield	43oh	Arduino Motor Control	Adafruit Motor Driver	adafruit.com	1	\$24.99
Power Supply	Circuit Specialists	12V Power Supply (150W)	PS1-150W-12	Circuit Specialists.com	1	\$26.85
Development workstation	(Manufacturer is irrelevant)	Development workstation computer	(Provided by group)	(Provided by group)	1	\$0.00
Custom PCB Board	OSHPark	Full spec 2-layer PCB	N/A	4pcb.com	1	\$33.00

The following table is a list of software parts that were necessary for development of the Universal Circuit Fabricator and its subsystems. These parts are required to build a working prototype of the design before a building a solid, final product

Table 49 – Software Development Bill of Materials

Item	Manufacturer	Description	Part Number	Source	Qty	Price
Microcontroller code development software	Arduino	Arduino Development Environment	Arduino 0021 - Windows	arduino.cc	1	\$0.00
Data collection and processing IDE	Microsoft	Microsoft Visual Studio C++ Express Edition	Visual C++ Express Edition 2010	Microsoft.com	1	\$0.00
Code Composer Studio	Texas Instruments	MSP430 Development Environment	Code Composer Studio Version 6	ti.com/ccstudio	1	\$0.00
Gcodesender	N/A	Open library software to send Gcode to UCF	N/A	gcodesender.com	1	\$0.00
Development workstation	(Manufacturer is irrelevant)	Development workstation computer	(Provided by group)	(Provided by group)	1	\$0.00

The following is a list of the inks that will be tested to select the best option for development purposes. After an ink is selected, it will be incorporated into the prototype and thoroughly tested.

Table 50 – Conductive Ink Bill of Materials

Inks	Price
Gallium/Indium Ink	\$75
Copper Sulfate Ink	\$75
Charcoal – Carbon Ink	\$10
Silver Nitrate Ink	\$47
Silver Acetate Ink	\$77

12.2 FINANCING

Financing for the project will be fulfilled by the sponsorship received from Boeing, Inc. The sponsorship will be used to purchase the necessary components for the Universal Circuit Fabricator. Some of the funding will also be used to purchase different kinds of conductive inks. These inks will be tested and the one with the best performance in relation to the cost will be selected. Another portion of the funds will be used to purchase the structure in which the Universal Circuit Fabricator will be built upon.

Once the prototype is completed, all of the breakout boards will be implemented into a final design PCB. The PCB manufacturer chosen for the UCF was OshPark as it provided great service for a reasonable price, as well as providing three copies of the printed circuit board in case an error in assembly occurred.

13 Project Milestone Schedule

The Universal Circuit Fabricator Senior Design Project Schedule stretches a span of two semesters. The first semester (August – December) involves project planning, budgeting, and documentation. The second semester (January – May) involves purchasing parts, building, and troubleshooting the prototype.

The Project Plan includes a research phase, subsystem development phase, subsystem integration phase, and prototyping testing phase. Each phase is broken down into sub-phases and tasks that have a specific duration.

The Project Plan below was created before design was actually begun, and within Senior Design II, the actual milestones and schedule was vastly different from the original plan. However, the group tried to maintain the milestones for a successful completion date.

	Task Name	Duration	Start	Finish	Predecessors
1	Research	17 days	Mon 1/5/15	Wed 1/21/15	
2	Ink Type Research	14 days	Mon 1/5/15	Sun 1/18/15	
3	Gallium/Indium Ink Production	13 days	Mon 1/5/15	Sat 1/17/15	
4	Gallium/Indium Ink Conductivity Testing	1 day	Sun 1/18/15	Sun 1/18/15	3
5	Copper Sulfate Ink Production	13 days	Mon 1/5/15	Sat 1/17/15	
6	Copper Sulfate Ink Conductivity Testing	1 day	Sun 1/18/15	Sun 1/18/15	5
7	Silver Nitrate Ink Production	13 days	Mon 1/5/15	Sat 1/17/15	
8	Silver Nitrate Ink Conductivity Testing	1 day	Sun 1/18/15	Sun 1/18/15	7
9	Reduced Charcoal Ink Production	13 days	Mon 1/5/15	Sat 1/17/15	
10	Reduced Charcoal Ink Conductivity Testing	1 day	Sun 1/18/15	Sun 1/18/15	9
11	Graphite Shaving Ink Production	13 days	Mon 1/5/15	Sat 1/17/15	
12	Graphite Shaving Ink Conductivity Testing	1 day	Sun 1/18/15	Sun 1/18/15	11
13	Graphite Glue Ink Production	13 days	Mon 1/5/15	Sat 1/17/15	
14	Graphite Glue Ink Conductivity Testing	1 day	Sun 1/18/15	Sun 1/18/15	13
15	Silver Acetate Ink Production	13 days	Mon 1/5/15	Sat 1/17/15	
16	Silver Acetate Ink Conductivity Testing	1 day	Sun 1/18/15	Sun 1/18/15	15
17	Ink Shield Research	7 days	Thu 1/8/15	Wed 1/14/15	
18	Gallium/Indium Ink Flow Test	1 day	Thu 1/8/15	Thu 1/8/15	
19	Copper Sulfate Ink Flow Test	1 day	Fri 1/9/15	Fri 1/9/15	18
20	Silver Nitrate Ink Flow Test	1 day	Sat 1/10/15	Sat 1/10/15	19
21	Reduced Charcoal Ink Flow Test	1 day	Sun 1/11/15	Sun 1/11/15	20
22	Graphite Shaving Ink Flow Test	1 day	Mon 1/12/15	Mon 1/12/15	21
23	Graphite Glue Ink Flow Test	1 day	Tue 1/13/15	Tue 1/13/15	22
24	Silver Acetate Ink Flow Test	1 day	Wed 1/14/15	Wed 1/14/15	23
25	Substrate Type Research	7 days	Thu 1/15/15	Wed 1/21/15	
26	Printer Paper Feasibility	1 day	Thu 1/15/15	Thu 1/15/15	24
27	Photo Paper Feasibility	1 day	Fri 1/16/15	Fri 1/16/15	26
28	PET Transparency Feasibility	1 day	Sat 1/17/15	Sat 1/17/15	27
29	Canvas Feasibility	1 day	Sun 1/18/15	Sun 1/18/15	28
30	FR4 Silicon Substrate Feasibility	1 day	Mon 1/19/15	Mon 1/19/15	29
31	Glass Feasibility	1 day	Tue 1/20/15	Tue 1/20/15	30
32	Acrylic Film Feasibility	1 day	Wed 1/21/15	Wed 1/21/15	31
33	Milestone: Selection of Ink + Substrate Combination	0 days	Wed 1/21/15	Wed 1/21/15	32

Figure 35 – Project Plan including Milestones (continued on next page)

	Task Name	Duration	Start	Finish	Predecessors
34	[-] Subsystem Development	186 days	Thu 9/11/14	Sun 3/15/15	
35	[-] Inkjet Cartridge Control Subsystem	149 days	Thu 9/11/14	Fri 2/6/15	
36	Acquire InkShield Part	15 days	Thu 9/11/14	Thu 9/25/14	
37	Assemble and Program Arduino with InkShield	7 days	Fri 9/26/14	Thu 10/2/14	36
38	Arduino Conductive Ink Strait Line Test	1 day	Thu 1/22/15	Thu 1/22/15	33
39	Convert Source Code into TI CCS via Energia	7 days	Fri 1/23/15	Thu 1/29/15	38
40	Assemble and Program MSP430 with InkShield	7 days	Fri 1/30/15	Thu 2/5/15	39
41	MSP430 Conductive Ink Strait Line Test	1 day	Fri 2/6/15	Fri 2/6/15	40
42	Milestone: Functioning Printhead using MSP430	0 days	Fri 2/6/15	Fri 2/6/15	41
43	[-] Data Input and Processing Subsystem	28 days	Thu 1/22/15	Wed 2/18/15	
44	Acquire Parts	14 days	Thu 1/22/15	Wed 2/4/15	33
45	Integrate SD Card reader with MSP430	7 days	Thu 2/5/15	Wed 2/11/15	44
46	Decode bitmap from SD Card	7 days	Thu 2/12/15	Wed 2/18/15	45
47	Integrate Capacitive Touch Sensor with MSP430	7 days	Thu 2/5/15	Wed 2/11/15	44
48	Create bitmap from capacitive touch input	7 days	Thu 2/12/15	Wed 2/18/15	47
49	Milestone: Functionality of Data Input Subsystem	0 days	Wed 2/18/15	Wed 2/18/15	48
50	[-] LCD Graphical User Interface Subsystem	53 days	Thu 1/22/15	Sun 3/15/15	
51	Acquire Parts	14 days	Thu 1/22/15	Wed 2/4/15	33
52	Integrate LCD screen with MSP430	7 days	Thu 2/5/15	Wed 2/11/15	51
53	Display test image	1 day	Thu 2/12/15	Thu 2/12/15	52
54	Develop GUI enviroment	7 days	Fri 2/13/15	Thu 2/19/15	53
55	Program MSP430 to run GUI	7 days	Fri 2/20/15	Thu 2/26/15	54
56	Display top menu of GUI on LCD	2 days	Fri 2/27/15	Sat 2/28/15	55
57	Integrate Capacitive Touch / Input Sensors with MSP430	7 days	Sun 3/1/15	Sat 3/7/15	56
58	Navigate GUI menus using Input buttons	5 days	Sun 3/8/15	Thu 3/12/15	57
59	Verify Capacitive Touch design functionality in GUI	3 days	Fri 3/13/15	Sun 3/15/15	58
60	Milestone: Functionality of GUI Subsystem	0 days	Sun 3/15/15	Sun 3/15/15	59
61	[-] Motor Control Subsystem	31 days	Thu 1/22/15	Sat 2/21/15	
62	Acquire Parts	14 days	Thu 1/22/15	Wed 2/4/15	33
63	Integrate with MSP430	14 days	Thu 2/5/15	Wed 2/18/15	62
64	Verify motor step precision	3 days	Thu 2/19/15	Sat 2/21/15	63
65	Milestone: Functionality of Motor Control Subsystem	0 days	Sat 2/21/15	Sat 2/21/15	64
66	[-] Annealing Subsystem	24 days	Thu 1/22/15	Sat 2/14/15	
67	Acquire Parts	14 days	Thu 1/22/15	Wed 2/4/15	33
68	Integrate with Microcontroller	7 days	Thu 2/5/15	Wed 2/11/15	67
69	Verify Microcontroller regulates temperature	3 days	Thu 2/12/15	Sat 2/14/15	68
70	Milestone: Functionality of Annealing Subsystem	0 days	Sat 2/14/15	Sat 2/14/15	69
71	Milestone: End Subsystem Phase	0 days	Sun 3/15/15	Sun 3/15/15	42,49,60,65,70

Figure 36 – Project Plan including Milestones (continued on next page)

	Task Name	Duration	Start	Finish	Predecessors
72	▢ Subsystem Integration	206 days	Fri 9/5/14	Sun 3/29/15	
73	▢ PCB Order	199 days	Fri 9/5/14	Sun 3/22/15	
74	Design PCB Layout	5 days	Mon 3/16/15	Fri 3/20/15	71
75	Order PCB	2 days	Sat 3/21/15	Sun 3/22/15	74
76	Milestone: Arrival of custom PCB	0 days	Fri 9/5/14	Fri 9/5/14	
77	Integrete Subsystems using MSP430 Launchpad	7 days	Mon 3/16/15	Sun 3/22/15	71
78	Milestone: Functioning MSP430 Launchpad Prototype	0 days	Sun 3/22/15	Sun 3/22/15	77
79	Integrate Subsystems using custom PCB	7 days	Mon 3/23/15	Sun 3/29/15	76,78
80	Milestone: Functioning prototype on custom PCB	0 days	Sun 3/29/15	Sun 3/29/15	79
81	Milestone: End Integration Phase	0 days	Sun 3/29/15	Sun 3/29/15	80
82	▢ Prototype Testing	21 days	Mon 3/30/15	Sun 4/19/15	
83	Straight Line Continuity/Resistivity Test	2 days	Mon 3/30/15	Tue 3/31/15	81
84	Print 1k ohm resister	5 days	Wed 4/1/15	Sun 4/5/15	83
85	Switch + LED Circuit Design Test	7 days	Mon 4/6/15	Sun 4/12/15	84
86	Conductive touch pads feasibility	7 days	Mon 4/13/15	Sun 4/19/15	85
87	Milestone: Funtional Prototype	0 days	Sun 4/19/15	Sun 4/19/15	86

Figure 37 – Project Plan including Milestones

14 Final Project Summary / Conclusions

Circuit design has always gone hand in hand with Electrical Engineering. As a senior design group of electrical engineering seniors, circuit design has also correlated to the usage of breadboards, argued to be a tool that has grown archaic since its conception in the 1970s. Breadboards can be needlessly complicated, with hard to manage cable organization, a lack of visual appeal, and troubleshooting challenges that stem from the large amount of components that breadboards generally use.

The Universal Circuit Fabricator (UCF) serves to solve these problems by implementing an idea that removes the breadboard out of circuit design. Instead of dealing with complex wires and unorganized connections, the UCF is a machine that will instead use a conductive ink to print on a substrate to create a continuous circuit trace. With this project, hobbyists and engineers alike will no longer struggle to figure out how to translate their circuit design from their computer screen to their own physical design.

The UCF design resembles a typical inkjet printer, found in most homes. However, instead of ink, the cartridge and print head nozzles will contain a silver acetate compound that after being printed on a glass substrate will have low resistivity and be able to conduct current. A series of stepper motors will move the inkjet cartridge over the ink and continuously spray the ink on the glass substrate. After the entire circuit design has been printed, the ink has set and the nozzles have finished printing, the glass print bed will be heated, fusing the silver acetate compound into a continuous line that is capable of conducting current.

The UCF works by having a user input a circuit diagram bitmap from a popular bitmap creation software, like Microsoft Paint or their software of choice. The bitmap will be converted to a series of Gcode instructions that will be processed by an Atmel ATmega328P microcontroller that will take the data and transfer it to a series of subsystems.

First, the Inkjet Cartridge System will take the data path from the microprocessor and start to enable the HP C6602 Ink Cartridge in order to print on the substrate. The microprocessor code will have instructions whether to start and stop spraying the inkjet print heads.

Within the Motor Control System, the microprocessor, will also relay instructions to the two x and y axis stepper motors that will determine the start and stop of the two motors. With two motors, one will be solely responsible for the x direction, and the other motor will only be responsible for movement in the y direction. Both motors will be attached to a ribbon and gantry system that will also hold a carriage that contains the inkjet cartridge. As the motors move from the instructions from the microprocessor, the inkjet cartridge will move along with it to ensure accuracy in the final circuit trace. After the whole image has been processed, the motors will move back to their original position.

Finally, after the preliminary circuit trace has been printed on the substrate, the annealing system will take over, and a hot plate will be used to heat the glass to a temperature of 200 degrees F. The heat will allow the silver acetate ink to fuse to the glass and create a continuous circuit capable of carrying current.

The final design from the UCF should create a positive effect on electrical engineering. By removing the complexities of the breadboard, circuit design will be more accessible to the general public. University classes could also be helped by allowing current electrical engineering students to learn circuit design without struggling with the complexities of breadboard design. It will allow them to use a circuit schematic that they have learned about or designed in class and print it directly for laboratory or design use. With the design of the UCF, it is hopeful that circuit design continues to evolve and be relevant within the modern technological world.

We found the process of creating the UCF was greatly rewarding, as we learned how to combine and use the knowledge we received over our many semesters at school to create something that we are proud to share with our class as well as the engineering department. Our experience in senior design also prepared us for careers within the engineering field. Specifically, we have learned how to work in a group and function within a deadline oriented environment. Additionally, we have gained experience working on a long-term, complex project that had multiple unforeseen variables. However, ultimately, we are proud of our work with the Universal Circuit Fabricator, and are satisfied with our accomplishment within Senior Design.

Appendix A: References

A.1 References

The following section contains reference material that was used in the development of the Universal Circuit Fabricator Project. Sources have been documented in the Bibliography section.

A.1.1 Bibliography

Gallium-Indium Ink Source

<http://spectrum.ieee.org/geek-life/hands-on/how-to-brew-your-own-conductive-ink>

Copper Sulfate Ink Sources

<https://www.youtube.com/watch?v=9iyRUBvd260>

https://www.youtube.com/watch?v=el6vg6l7Z_g

Silver Nitrate Ink Sources

<http://pubs.acs.org/doi/abs/10.1021/jp100326k>

<https://www.youtube.com/watch?v=dfNByi-rrO4>

Reduced Charcoal Ink Source

<http://www.instructables.com/id/1-DIY-Conductive-Ink/>

Graphite Shaving Ink Source

<https://www.youtube.com/watch?v=egMr9kk1uHk>

Graphite Glue Ink Source

<http://www.instructables.com/id/Make-Conductive-Glue-and-Glue-a-Circuit/?ALLSTEPS>

Silver Acetate Ink Sources

<http://jordanbunker.com/archives/41>

<https://www.youtube.com/watch?v=EBIqPS8boLI>

<http://pubs.acs.org/doi/abs/10.1021/ja209267c>

A.2 Permissions

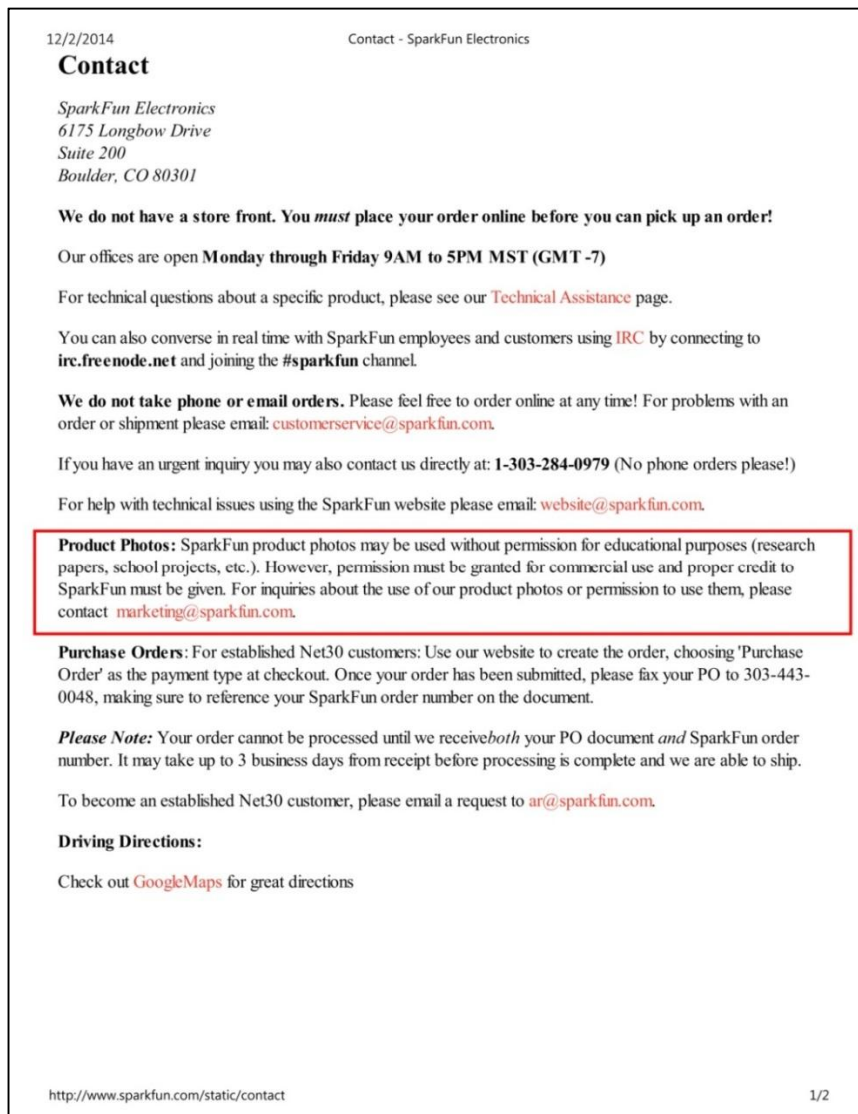
The following section contains information on licenses and permissions that were used in the development of this design document.

A.2.1 Figure Reprint Permissions

The following includes permissions, either emails or legal notices, for any reprinted figures, tables, or pictures taken from websites, datasheets, and other forms of media. Each image throughout the document has subtext denoting that it was reprinted with permission from the owner company. These are the permissions from the owners.

A.2.1.1 SparkFun

The following image, is a screenshot of the permission given by SparkFun to use any image on the website without permission for educational purposes.




A.2.1.2 Texas Instruments

The following images are screenshots of Texas Instruments Terms of Use. It states that any materials posted on their website/database, can be used for informational and non-commercial use.

12/2/2014

Texas Instruments - Terms of Use



[Samples & Purchase Cart](#) | [Contact Us](#) | [TI Worldwide: United States](#) | [my.TI Login](#)

[Products](#) | [Applications](#) | [Design Support](#) | [Sample & Buy](#)

All Searches Search by Keyword GO Search by Part Number GO

TI Home > Terms of Use

Site Terms of Use

The information, material and related graphics available on this site ("Materials") are provided by Texas Instruments Incorporated ("TI"). The following terms govern use of this site. By using this site you agree that you have read and understood these terms and agree to be bound by these terms, and to comply with all applicable laws and regulations regarding use of this site. If you do not agree to these terms, do not use this site.

TI operates this site from its offices within the United States. TI makes no representations that the Materials referenced on this site are appropriate or available for use in other areas of the world. Those who access this site from locations outside the United States are responsible for compliance with applicable local laws. Any claim relating to this site or use of this site will be governed by and interpreted in accordance with the laws of the State of Texas, without reference to its conflict-of-laws principles. Any dispute arising out of or related to your use of this site will be brought in, and you hereby consent to exclusive jurisdiction and venue in, the state and federal courts sitting in Dallas County, Texas. You agree to waive all defenses of lack of personal jurisdiction and forum non-conveniens and agree that process may be served in a manner authorized by applicable law or court rule.

TI reserves the right to make changes to this site and to these terms at any time. Any change in these terms will be prospective only, unless retroactive effect is legally required. Your continued use of this site will constitute your acceptance of any new or amended terms.

Use Restrictions

The Materials contained on this site are protected by copyright laws, international copyright treaties, and other intellectual property laws and treaties. Except as stated herein, these Materials may not be reproduced, modified, displayed or distributed in any form or by any means without TI's prior written consent.

TI grants permission to download, reproduce, display and distribute the Materials posted on this site solely for informational and non-commercial or personal use, provided that you do not modify such Materials and provided further that you retain all copyright and proprietary notices as they appear in such Materials. TI further grants to educational institutions (specifically K-12, universities and community colleges) permission to download, reproduce, display and distribute the Materials posted on this site solely for use in the classroom, provided that such institutions identify TI as the source of the Materials and include the following credit line: "Courtesy of Texas Instruments". Unauthorized use of any of these Materials is expressly prohibited by law, and may result in civil and criminal penalties. This permission terminates if you breach any of these terms and conditions. Upon termination you agree to destroy any Materials downloaded from this site.

Warranties and Disclaimers

TI intends for the Materials contained on this site to be accurate and reliable. These Materials may, however, contain technical inaccuracies, typographical errors or other mistakes. TI may make corrections or other changes to these Materials at any time. TI and its suppliers reserve the right to make corrections, modifications, enhancements, improvements and other changes to its products, programs and services at any time or to discontinue any products, programs, or services without notice.

THE MATERIALS ON THIS SITE ARE PROVIDED "AS IS". TI AND ITS RESPECTIVE SUPPLIERS MAKE NO REPRESENTATIONS ABOUT THE SUITABILITY OF THESE MATERIALS FOR ANY PURPOSE AND DISCLAIM ALL WARRANTIES AND CONDITIONS WITH REGARD TO THESE MATERIALS, INCLUDING BUT NOT LIMITED TO, ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHT.

YOU ACKNOWLEDGE AND AGREE THAT THE APPLICATION NOTES, REFERENCE DESIGNS AND OTHER SUCH DESIGN MATERIALS INCLUDED HEREIN ARE PROVIDED AS AN EXAMPLE ONLY AND THAT YOU WILL EXERCISE YOUR OWN INDEPENDENT ANALYSIS AND JUDGMENT IN YOUR USE OF THESE MATERIALS. TI ASSUMES NO LIABILITY FOR YOUR USE OF THESE MATERIALS OR YOUR PRODUCT DESIGNS OR ANY APPLICATIONS ASSISTANCE PROVIDED BY TI.

TI DOES NOT WARRANT OR REPRESENT THAT ANY LICENSE, EITHER EXPRESS OR IMPLIED, IS GRANTED UNDER ANY PATENT RIGHT, COPYRIGHT, MASK WORK RIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT OF TI COVERING OR RELATING TO THESE MATERIALS OR ANY COMBINATION, MACHINE, OR PROCESS TO WHICH THESE MATERIALS RELATE OR WITH WHICH THESE MATERIALS MAY BE USED.

USE OF THE INFORMATION ON THIS SITE MAY REQUIRE A LICENSE FROM A THIRD PARTY UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF THAT THIRD PARTY, OR A LICENSE FROM TI UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF TI.

Limitation of Liability

ti.com/corp/docs/legal/termsfuse.shtm...

1/2

12/2/2014

Texas Instruments - Terms of Use

IN NO EVENT SHALL TI OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER, INCLUDING BUT NOT LIMITED TO, DAMAGES RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION RESULTING FROM USE OF THIS SITE OR ARISING OUT OF THE USE OR PERFORMANCE OF THE MATERIALS AVAILABLE ON THIS SITE, REGARDLESS OF WHETHER TI OR AN AUTHORIZED TI REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Specific Notice Regarding Software Available on This Site

Any software that is made available to download from this site ("Software") is copyrighted. Use of this Software is governed by the terms of the license agreement, if any, that accompanies or is included with such Software ("License Agreement"). A user will be unable to install any Software that is accompanied by or includes a License Agreement before first agreeing to the License Agreement terms. Any reproduction or redistribution of the Software not in accordance with the License Agreement is expressly prohibited, and may result in civil and criminal penalties.

WITHOUT LIMITING THE FOREGOING, COPYING OR REPRODUCTION OF THE SOFTWARE TO ANY OTHER LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION IS PROHIBITED, UNLESS SUCH REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PERMITTED BY THE LICENSE AGREEMENT ACCOMPANYING SUCH SOFTWARE. FOR YOUR CONVENIENCE, TI MAY MAKE AVAILABLE ON THIS SITE OR IN ITS SOFTWARE PRODUCTS, TOOLS AND UTILITIES FOR USE OR DOWNLOAD. TI DOES NOT MAKE ANY ASSURANCES WITH REGARD TO THE ACCURACY OF THE RESULTS OR OUTPUT THAT DERIVES FROM SUCH USE OF ANY SUCH SOFTWARE PRODUCTS, TOOLS AND UTILITIES.

Specific Notice Regarding Links to Third Party Sites

Certain links provided herein permit you to leave this site and enter non-TI sites. These linked sites are not under TI's control. TI is not responsible for the contents of any linked site or any changes or updates to such sites. TI is providing these links to you only as a convenience. The inclusion of any link does not imply endorsement by TI of any linked site.

TI'S PUBLICATION OF INFORMATION REGARDING THIRD-PARTY PRODUCTS OR SERVICES DOES NOT CONSTITUTE AN ENDORSEMENT REGARDING THE SUITABILITY OF SUCH PRODUCTS OR SERVICES OR A WARRANTY, REPRESENTATION OR ENDORSEMENT OF SUCH PRODUCTS OR SERVICES, EITHER ALONE OR IN COMBINATION WITH ANY TI PRODUCT OR SERVICE.

Linking to this site is subject to [TI's Linking Policy](#).

Specific Notice Regarding Products Purchased Over This Site

Unless otherwise specified, products purchased from this site are subject to TI's Standard Terms and Conditions of Sale, which should be reviewed carefully before placing an order. To review TI's Standard Terms and Conditions of Sale:

[Semiconductor Products](#)
[Software Products](#)

Privacy Policy

Use of this site is subject to [TI's Privacy Policy](#).

Products | **Applications** | **Design Support** | **Sample & Buy** 

TI Worldwide | About TI | Contact Us | Investor Relations | News Center | Corporate Citizenship | Careers | Tags | my.TI Login | All Searches | Site Map
© Copyright 1995-2010 Texas Instruments Incorporated. All rights reserved. Trademarks | Privacy Policy | Terms of Use

A.3 Screenshot Permissions

The following includes permissions, either emails or legal notices, for any screenshots taken of software being used in the project. Since these are screenshots by the group of software that is obtained by purchase or through free use licensing, the legal guidelines are different from a reprinted figure or table. Each image throughout the document has subtext denoting that it is a screenshot taken with permission from the owner company. These are the permissions from the owners.

A.3.1 Arduino

The following images contain the Frequently Asked Questions on the Arduino website. One of the questions asks about Arduino's open-source. It states that the Arduino software is open-source and is registered under the LGPL.

12/2/2014 Arduino - FAQ

search

Buy Download Getting Started Learning Reference Hardware FAQ Blog » Forum » Playground »

Frequently Asked Questions

What is an Arduino? Glad you asked, we have a great introduction page on Arduino, [click here to read it](#).

What do you mean by open-source hardware? Open-source hardware shares much of the principles and approach of free and open-source software. In particular, we believe that people should be able to study our hardware to understand how it works, make changes to it, and share those changes. To facilitate this, we release all of the original design files (Eagle CAD) for the Arduino hardware. These files are licensed under a Creative Commons Attribution Share-Alike license, which allows for both personal and commercial derivative works, as long as they credit Arduino and release their designs under the same license.

The Arduino software is also open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL.

How can I get an Arduino board? You can buy an Arduino board from one of the distributors listed on the [buy](#) page. If you'd prefer to build your own, see the [Arduino Single-Sided Serial board](#), which can be easily etched and assembled.

Who makes Arduino boards? Most of the official Arduino boards are manufactured by SmartProjects in Italy. The Arduino Pro, Pro Mini, and LilyPad are manufactured by SparkFun Electronics (a US company). The Arduino Nano is manufactured by Gravitech (also a US company).

Which are the official Arduino boards? The official Arduino boards are the ones listed on the [hardware page](#): the Duemilanove, Nano, Mega, Bluetooth (BT), LilyPad, Mini, Pro, Pro Mini, and a few older models, along with the Ethernet, XBee, motor, and prototyping shields. These are boards whose manufacturers work with the Arduino team to ensure a good user experience, compatibility with the Arduino software, and a quality product. In return for their status as official boards, the manufacturers pay a licensing fee to the Arduino team to support the further development of the project. In general, we try to restrict use of the name "Arduino" to the official boards. If you find a product under a different name but described as "Arduino compatible", it's probably not an

<http://arduino.cc/en/Main/FAQ> 1/3

12/2/2014

Arduino - FAQ

official board and doesn't fund continued work on the project.

I want to design my own board; what should I do?

The reference designs for the Arduino boards are available from the [hardware](#) page. They're licensed under a Creative Commons Attribution Share-Alike license, so you are free to use and adapt them for your own needs without asking permission or paying a fee. If you're looking to make something of interest to the community, we'd encourage you to discuss your ideas on the [hardware development forum](#) so that potential users can offer suggestions.

What should I call my boards?

If you're making your own board, come up with your own name! This will allow people identify you with your products and help you to build a brand. Be creative: try to suggest what people might use the board for, or emphasize the form factor, or just pick a random word that sounds cool. "Arduino" is a trademark of Arduino team and should not be used for unofficial variants. If you're interested in having your design included in the official Arduino product line, please see the [So you want to make an Arduino](#) document and contact the Arduino team. While unofficial products should not have "Arduino" in their name, it's okay to describe your product in relation to the Arduino project and platform. Here are a few guidelines that explain which uses we consider reasonable. Not okay:

- ✦ Arduino Xxxxxx
- ✦ Xxxxxx Arduino
- ✦ Arduino Compatible Xxxxxx - use "Xxxxxx (Arduino-Compatible)" instead

Okay:

- ✦ Xxxxxx for Arduino - products that work with official Arduino boards (e.g. shields or kits)
- ✦ Xxxxxx (Arduino-Compatible) - variations and clones which are software and hardware compatible

Note that while we don't attempt to restrict uses of the "duino" suffix, its use causes the Italians on the team to cringe (apparently it sounds terrible); you might want to avoid it. (It's also trademarked by a Hungarian company.)

Can I build a commercial product based on Arduino?

Yes, with the following conditions:

- ✦ Physically embedding an Arduino board inside a commercial product does not require you to disclose or open-source any information about its design.
- ✦ Deriving the design of a commercial product from the Eagle files for an Arduino board requires you to release the modified files under the same Creative Commons Attribution Share-Alike license. You may manufacture and sell the resulting product.
- ✦ Using the Arduino core and libraries for the firmware of a commercial product does not require you to release the source code for the firmware. The LGPL does, however, require you to make available object files that allow for the relinking of the firmware against updated versions of the Arduino core and libraries. Any modifications to the core and libraries must be released under the LGPL.
- ✦ The source code for the Arduino environment is covered by the GPL, which requires any modifications to be open-sourced under the same license. It does not prevent the sale of derivative software or its inclusion in commercial products.

<http://arduino.cc/en/Main/FAQ>

2/3

12/2/2014

Arduino - FAQ

In all cases, the exact requirements are determined by the applicable license. Additionally, see the previous question for information about the use of the name "Arduino".

How can I run the Arduino IDE under Linux?

See instructions [for Ubuntu Linux](#), [for Debian Linux](#), [for Gentoo Linux](#), [for Linux](#), or [for Linux on PPC](#). This [this forum thread](#) has more information. Or, you can [use Arduino from the command line](#), and not have to install Java.

Can I program the Arduino board in C?

In fact, you already are; the Arduino language is merely a set of C/C++ functions that can be called from your code. Your sketch undergoes minor changes (e.g. automatic generation of function prototypes) and then is passed directly to a C/C++ compiler (avr-g++). All standard C and C++ constructs [supported by avr-g++](#) should work in Arduino. For more details, see the page on the [Arduino build process](#).

Can I use a different IDE to program the Arduino board?

It is possible to compile programs for the Arduino using other build tools (e.g. Makefiles and/or AVR Studio). You'll need to configure these to link against the appropriate files in the Arduino core libraries. See the description of the Arduino [build process](#).

Can I use an Arduino board without the Arduino software?

Sure. It's just an AVR development board, you can use straight AVR C or C++ (with avr-gcc and avrdude or AVR Studio) to program it.

Can I use the Arduino software with other AVR boards?

Yes, although it may require some modifications to the Arduino core libraries. See the [porting page](#) in the Arduino Google Code project for details.

Where is the troubleshooting section?

These questions have moved to the [troubleshooting](#) section of the Arduino [guide](#).

Share |

[©Arduino](#) | [Edit Page](#) | [Page History](#) | [Printable View](#) | [All Recent Site Changes](#)

<http://arduino.cc/en/Main/FAQ>


3/3

A.3.2 Energia

The following image contains the License deed that is used by energia.nu. It states that the materials posted on their website/database can be used freely.


12/3/2014

Creative Commons — Attribution-ShareAlike 3.0 Unported — CC BY-SA 3.0




[Creative Commons](#)
Creative Commons License Deed
Attribution-ShareAlike 3.0 Unported (CC BY-SA 3.0)


This is a human-readable summary of (and not a substitute for) the [license](#).
[Disclaimer](#)



You are free to:




Share — copy and redistribute the material in any medium or format




Adapt — remix, transform, and build upon the material
for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:



Attribution — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.



ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the [same license](#) as the original.

No additional restrictions — You may not apply legal terms or [technological measures](#) that legally restrict others from doing anything the license permits.

Notices:

You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable [exception or limitation](#).

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as [publicity, privacy, or moral rights](#) may limit how you use the

<http://creativecommons.org/licenses/by-sa/3.0/>

1/2