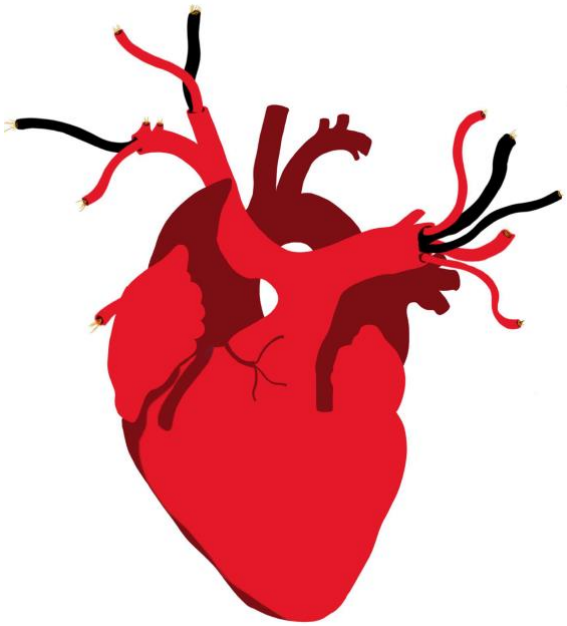
Heart Bytes



Design Report 4

12/9/2022

Eisa Alyaseen - eha33@nau.edu

Alex Anderson - amt646@nau.edu

Abdulrahman Aziz - aaa797@nau.edu

Introduction:

The Heart Bytes are an electrical engineering capstone team that worked alongside a mechanical engineering team to create a low-force stent crimping machine for W.L. Gore & Associates. W.L. Gore provided both of the teams with a total of $3,000 to create the machine given a list of customer requirements that should be met. One of the specific requirements that defined the fundamentals of the machine is that it must utilize an iris crusher mechanism to crimp stents. The mechanical engineering team was responsible for creating the iris crusher mechanism and the base for the machine. The Heart Bytes were responsible for creating the electrical system that would allow the machine to be functional.

Stents are medical devices that are inserted into patients' arteries to ensure that the vessels do not collapse or clog up. Low-force stent crimping machines crimp the stents before surgery to get the stent to the correct diameter. After the stents are put in the low-force stent crimping machine, the stents are placed into a high-force stent crimping machine to finish the crimp. Since the stent crimper is a machine that is a medical device, the team had to meet higher standards to ensure the safety of the machine. The team had to meet various ANSI and OSHA standards for medical devices. Included with these is that the team had to follow an iterative design process.

Several stent crimping machines on the market utilize similar mechanics, but one of the goals of the team was to produce one significantly under the market cost. The team went significantly under budget, so this goal was a success. The more expensive alternatives are more accurate and precise than the one created, but the team is still pleased with the final product. The machine consisted of an Arduino Mega microcontroller connected to a stepper motor and driver board, two different sensors, and a 7” touchscreen display.

This capstone project has been a massive learning experience for each of the members of the team. The team has learned how to go through the engineering process and figured out how to work better as a team. After putting in a year’s worth of work, the team has to be finished with the project for class even though there are more steps the team would have liked to complete to create a more finished product.

Requirements:

The sponsor from W.L. Gore & Associates provided the team with a list of customer requirements for the stent crimping machine. Using these requirements, the team created in-depth engineering requirements to ensure that the project met the customer's demands and make sure that the project would succeed.

The customer requirements were as follows:

1. The stent crimper must utilize an iris-shaped crushing mechanism
2. Radial force and diameter readouts after crimping
3. User inputs to control the diameter and radial force of the stent
4. The machine must meet relevant OSHA and ANSI standards
5. A working model of an Endovascular stent crimper
6. A reliable and precise design

The corresponding engineering requirements were:

1. The device will use an iris crusher designed by the ME team
2. The team will use a touchscreen display to show measurements
3. The touchscreen display will take user inputs for setting measurements
4. Include an emergency stop button and warning labels
5. Iris crimping range from 5mm to 50mm and exert a max force of 132.94N
6. Will use precise sensors to determine the measurements for diameter and radial forces with up to 1% accuracy

The team has met a substantial amount of the engineering requirements for the project but struggled to complete the last two. The stent crimping machine utilized an iris crusher that was designed by the mechanical engineering team. This is a success for both the 1st customer and engineering requirements. The team was able to get the touchscreen display to work and it outputs the readouts for the measurements of radial force and diameter. This is a success for both the 2nd customer and engineering requirements. The touchscreen display also took inputs to control the diameter and radial force of the stent. This is a success for the 3rd customer and engineering requirements. The machine had an emergency stop button and proper warning labels. This is a success for the 4th customer and engineering requirements. The range of the iris went from 2 - 62 mm and the maximum force is unknown, but it is known it is too low. This is a moderate success for the 5th customer and engineering requirements. The machine does not use accurate sensors. There is a high degree of error in both of the sensors. The team was not able to meet the 6th customer and engineering requirements.

Prototypes/Design Process:

The Heart Bytes team was formed at the beginning of February 2022. The team was assigned the task of creating the low-force stent crimping machine for W.L. Gore & Associates as soon as the team was assembled. By February 17th, the team had assigned roles for the team and compiled a list of each of the team members' abilities. During February, the team started to meet with the partnering mechanical engineering team and the project sponsors to get a better understanding of what was expected of the team. During this time, the team collected a list of the customers' requirements and created a list of engineering requirements that must be met to call the project a success.

After this point, the team started work on creating the stent crimping machine. The first thing the team had to decide would be how to control all of the electronic components of the machine. The team chose to go with the Arduino Mega microcontroller board because the board had 54 digital pins and 16 analog input pins. The team had not decided on the rest of the hardware for the project so the required number of pins was unknown, but the team was confident that the high number of pins on the Arduino Mega would be enough. The team was also familiar with the software for Arduino due to a prior class using Arduino boards. Now that the project has concluded, it can be stated that the Arduino Mega board was the correct microcontroller to use for the project.

After deciding on the microcontroller, the team started researching what type of graphical user interface should be used as well as what sensors and motor should be used. The team had initially decided to go with a series of seven-segment displays to display the measurements for radial force and diameter. The sensors and motors that would be used for the project would be determined at a future date.

The team then started their individual prototypes. There will be three subsections below to describe each of the team members' prototypes.

Eisa Alyaseen's Prototype:

The purpose of this prototype was to determine how to control the stepper motor that would be needed for the project. The stepper motor would be needed to rotate a shaft that will control the iris crushing mechanism's diameter. Eisa used an Arduino Uno board connected to an Elegoo byj48 stepper motor and a corresponding stepper motor driver board. Eisa ran into compatibility errors between the Arduino and the motor and was not able to get the motor to function.

Alex Anderson's Prototype:

The purpose of this prototype was to start working on the code that would be needed for the project. The included features for this prototype included: working user inputs to control the diameter and radial force of the stent, control of the stepper motor, and an emergency stop button. Future support for sensors and a GUI were added through various code stubs. The controls for the prototype were 4 push buttons to set the diameter and radial force, and an additional push button as an emergency stop. There were two potentiometers to simulate analog sensors. The stepper motor used for this prototype was the Elegoo byj48 stepper motor. The code written was functional but had to be rewritten at later stages of the project to be compatible with future parts. This prototype was successful because it refamiliarized the team with the Arduino coding language.

Abdulrahman Aziz's Prototype

The purpose of this prototype was to determine which sensors the team should use for building the stent crimping machine and how they should be used to get accurate results. At this point, it was planned that the distance sensor for determining the diameter of the stent would be inside of the iris mechanism. The sensors were purchased but there were compatibility issues between the sensors and the Arduino board.

After the individual prototypes, the team met to determine how the team should continue. Around this point, the mechanical engineering team suggested the team use a touchscreen display to take inputs and display outputs. The team researched touchscreen displays and determined it was possible. The requirements were rewritten to include a touchscreen display instead of a series of seven-segment displays. After discussing with the mechanical engineering team, it was determined the screen should be around 7". The team determined that the Nextion Intelligent Series 7" display would be the most suitable for the project. The touchscreen was compatible with Arduino and it contained an onboard microcontroller to process the display and the user inputs.

The team started learning how to use the touchscreen and how it communicates with Arduino boards. The code was rewritten during this process to add functionality to the touchscreen display. The team had still not chosen a motor or sensors for the project until there were only a few months left in the project due to difficulty in finding parts adequate for the project and difficulties in determining how the measurements for the machine needed to be taken. The motors and sensors chosen are discussed in detail in the next section.

Design:

Figure 1: Photograph of the Finalized Stent Crimping Machine

The low-force stent crimping machine was constructed of a steel base with the iris crimping mechanism extruding from the top. A touchscreen display and emergency stop buttons are also on the top face of the machine. The machine uses an extension cord coming out of the back of the machine to provide power for the machine. The base and iris crimping mechanism were designed by the mechanical engineering team. The electronics of the machine consisted of an Arduino Mega microcontroller connected to a stepper motor, a couple of sensors, and a touchscreen display. To determine the diameter of the stent, a rotary encoder was used. For determining the radial force applied to the stent, a force-sensitive resistor was used. The motor used for the project was initially a NEMA 23 stepper motor and has been changed to a NEMA 34 stepper motor in the final week of the project. To provide grounding for the steel base, a grounding wire was plugged into the extension cord.

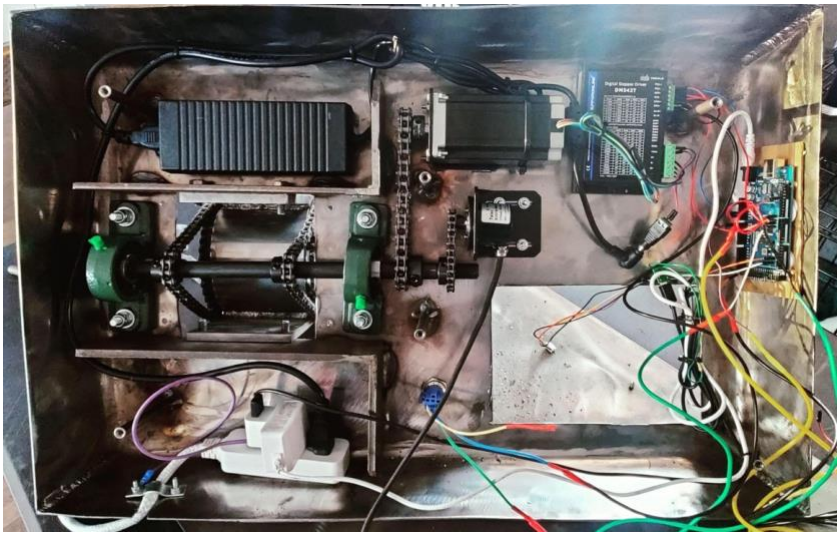


Figure 2: Underside of Base Revealing Final Wiring

At the center of the design for the electrical system was the Arduino Mega microcontroller. The board performed all of the processing for the machine. The code for the Arduino is in the appendix for reference. The board was attached to the side of the inside of the base using velcro strips. All of the wires on the board have been hot glued to the pin connectors to ensure that the wires do not fall out. The flowchart showing modeling the Arduino's processes can be seen in figure 3.

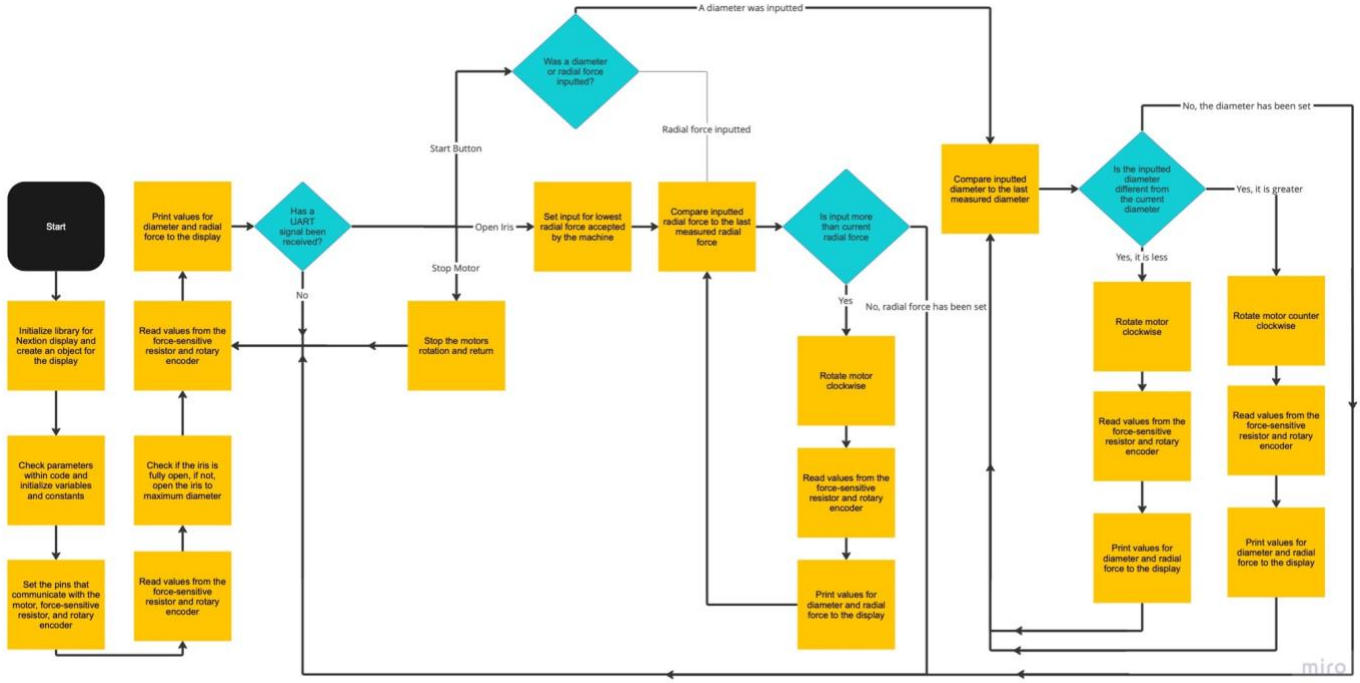


Figure 3: Flowchart Showing Arduinos Processes

Figure 4: Wiring for the Nextion Display

The touchscreen display used for the project was a Nextion 7" intelligent series display. The display contained a microcontroller that processed the display including the touch inputs and

screen updates. The inputs and outputs for the screen are handled by a 4-pin UART connection and a micro SD card slot. The SD slot is used to update the software on the touchscreen display. The UART connection is used to power the screen and transfer data to the Arduino. The 5V and ground wires of the UART connector are connected to a power board that takes power from a 5V 1A micro USB cable. The RX and TX pins are connected to the TX and RX pins on the Arduino.

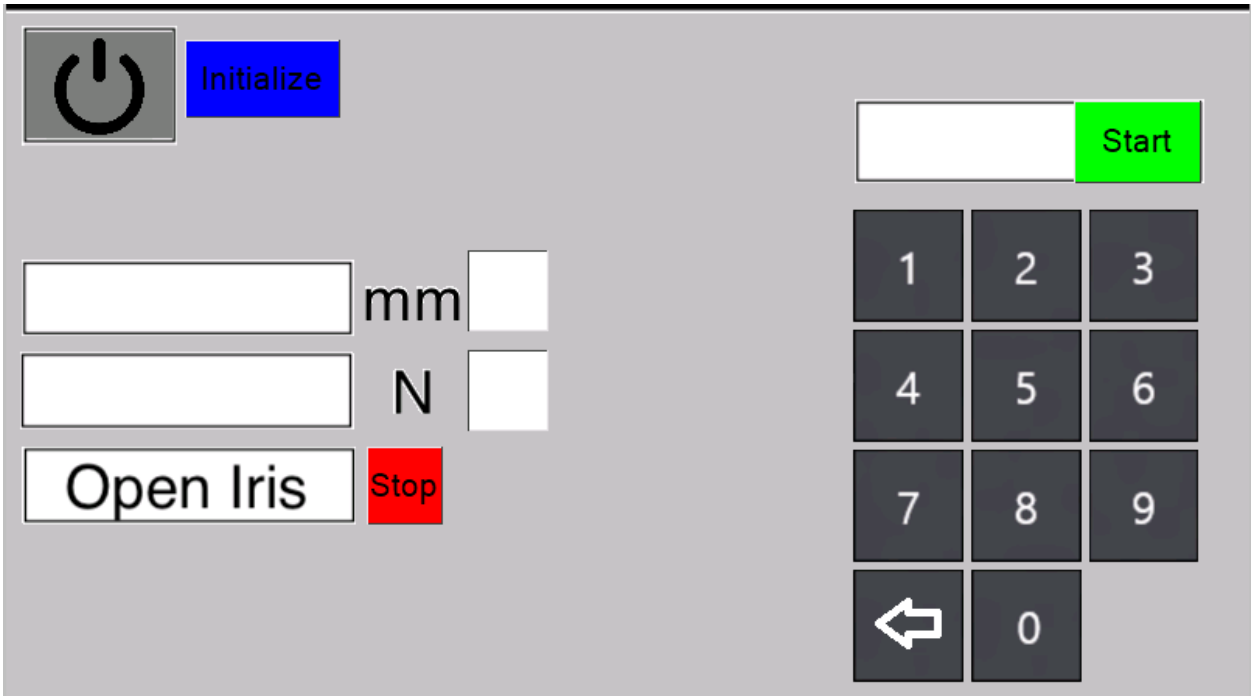


Figure 5: Final Graphical User Interface

The touchscreen display was coded using the Nextion Screen editing software. The power button turns off the display and it is woken up when the screen is touched again. The boxes next to the units select which unit the user is going to input. Once the user selects the units, the measurements the user wants the stent to be crimped to can be inputted into the number pad on the right side of the screen. The green button starts the crimping process. The blue button initializes the machine and fully opens the iris. The open iris button opens the iris up as well. The red button stops the motor from turning.

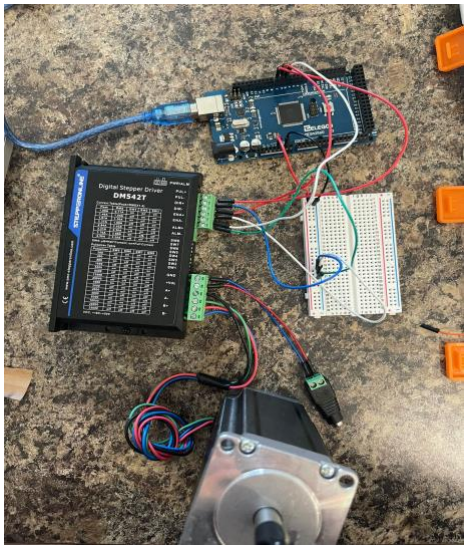


Figure 6: Wiring for the NEMA 23 Stepper motor and DM542T Board

The motor used initially for the project was a NEMA 23 StepperOnline stepper motor. It was driven by the DM542T stepper motor driver board. In the final stages of the project, the team switched the stepper motor to a NEMA 34 StepperOnline motor. The new motor is driven by the DM860T stepper motor driver board. The change to a new stepper motor occurred because the original motor was not able to provide enough torque to crimp stents to the lowest specified diameter. There is a gear on the shaft of the stepper motor that has a chain wrapped around it. The chain is also attached to another shaft that has several gears on it. The shaft rotating controls the iris crimping mechanism.

Figure 7: Wiring for the Taiss Rotary Encoder

The rotary encoder also is attached to the shaft via a chain. The rotary encoder used initially was a Taiss rotary encoder with 600 pulses per rotation. The rotary encoder was a digital sensor that used two digital input pins on the Arduino to get measurements. The sensor sent

pulses to the Arduino every time a change in angle was detected. The shaft only rotates around 40 degrees resulting in only 66 pulses being used to determine the diameter. Since the range of diameters the stent crimper can crimp to is from 62 to 2mm, the team had thought having around one pulse per millimeter changed would be adequate. This sensor was highly susceptible to vibrations and the vibrations of the motor caused the sensor to have an error, so the team determined they needed more than one pulse per unit change in millimeters. The team switched it to a Jeanoko rotary encoder with 1024 pulses per rotation which should have a higher build quality than the previous sensor. The new rotary encoder will allow the team to use 113 pulses to determine the diameter of the stent. Now that the team has double the amount of pulses to work with, the team is hoping there will be fewer errors. This part has not been put into the machine nor tested as of writing this.

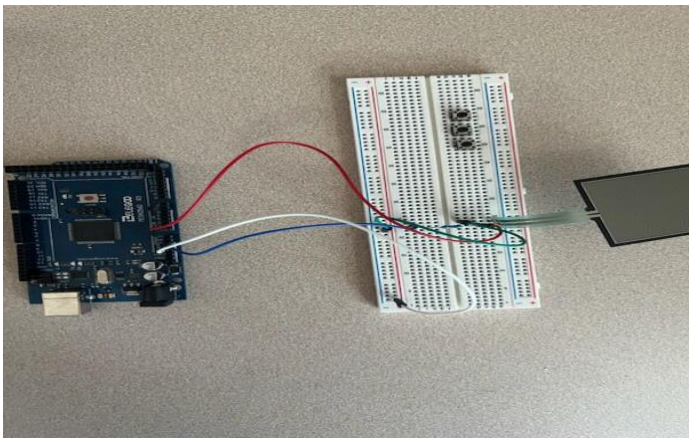


Figure 8: Wiring for the Force Sensitive Resistor

The force-sensitive resistor used is the Pololu 1.5” square force-sensitive resistor. The FSR is an analog sensor with a 5V input and a 10 kilo Ohm pulldown resistor. The resistor is attached between two of the fins on the iris crimping mechanism. The range of forces the sensor can read is from 0.2 N to 20 N force. Due to disagreements about how to implement the sensor with the mechanical engineering team, the force-sensitive resistor is not able to accurately read the radial forces applied to the stent. The force-sensitive resistor is only used as a limit switch while initializing the machine. The readouts of radial force that are printed come from an equation that the mechanical engineering team created that relates the diameter of a specific material to how much force would need to be applied for that specific material.

The emergency stop button for the project is attached between the Arduino and the stepper motor driver board. The stop button cuts the flow of the high signal, stopping the motor. The high signal that was intended to be sent to the motor gets rerouted to the reset pin of the Arduino to reset the software.

Next Steps:

The next steps to be taken on the project are to implement the new stepper motor and rotary encoder into the machine and finish calibrating the sensors properly. The new stepper motor is a NEMA 34 stepper motor with 13Nm of force from StepperOnline. Since the previous motor was only a NEMA 23 stepper motor, the team had to select a new stepper motor driver board and power supply to drive the motor. The new driver board is the DM860T also from StepperOnline. Since both of these pieces of hardware require more power than the previous motor, a new 36V power supply has been ordered. The connections from the DM860T are the same as on the DM542T driver board that is currently being used, so the team is hopeful that the process to switch the motors out will have no trouble.

The new rotary encoder has an additional wire on it from the rotary encoder that is currently being used in the project. The team will have to determine what the wire is and how to alter the code to accept the inputs from the new rotary encoder.

The team would also have to look into different methods of determining the radial force applied to the stent. The fins of the iris crushing mechanism did not always touch the force-sensitive resistor, causing inaccurate radial force readings. The force-sensitive resistor was a good first step, but after implementing it into the machine, the team determined that there has to be a more accurate way to determine the radial force.

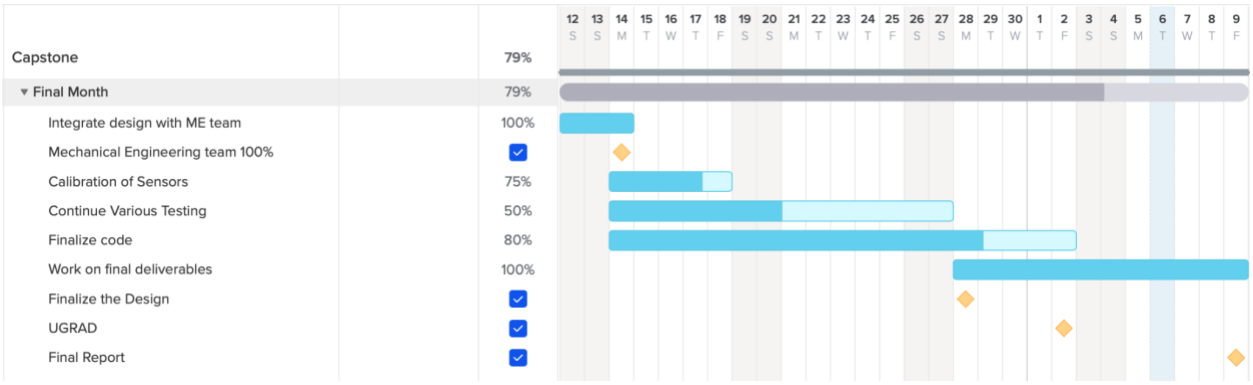


Figure 9: Final Gantt Chart for the project

As can be seen in the final Gantt chart for the project, the team was close to reaching 100% completion but was not able to achieve it. If the team was allowed to work for a single month longer, the team is confident that 100% completion of the Gantt chart would have been obtained.

Conclusion:

The team was successful in creating the stent crimping machine for the client. Although there are notable issues in the design regarding the accuracy of the sensors, the client has expressed with the team that they are pleased with the design, therefore the capstone project was a success. Working through this capstone project has been a learning experience for the team that showed the team how the engineering process works and how to better work on a team.

Working through the engineering design process for an entire year on a project showed the team where complications in the process come up and how to deal with them. The team underestimated the difficulty of brainstorming solutions to novel problems. For example, the team struggled with how to determine the diameter and radial force of the stent crimping machine without interfering with the processes of the machine. After spending hours brainstorming with the mechanical engineering team, we were able to find acceptable approximations for these measurements. The other big hurdle the team faced was in the testing step of the process. After the machine had been assembled, the team had not expected how long it would take to fix the bugs and implement all of the required features. Working through this process has made all of the team members significantly stronger engineers. If the team had one more month with the project, the team is confident that all of the requirements could be met.

The team is proud of the low-force stent crimping machine that was produced for the project. The overall design of the project was successful. The new motor will provide enough force for the machine. The new rotary encoder should improve the accuracy of the diameter reading and the team has determined that a force-sensitive resistor is not the best method for determining the radial force applied to the stent. The project was a success and taught the team several important lessons regarding the engineering design process.

Appendix:

Here is the final Arduino code used for the project:

```
#include "EasyNextionLibrary.h"

// Initialize initial values
double diameter = 62;
double radForce = 0;

// Create misc variables
String userInput = "";
int initFSRVolt;

// Define pins on arduino
#define resetPin 12
#define CLK 25 //(A/White)
#define DT 23 //(B/Green)
#define ena 5
#define dir 6
#define pul 7

// Create states for rotary encoder
int aLastState;
int aState;

// Initialize all flags
bool rotEncFlag = true;
bool diaFlag = false;
bool stopFlag = false;
bool motorFlag = false;

// Initialize pulse for motor
bool pulse = LOW;

// Initialize Nextion Screen as an object
EasyNex myNex(Serial1);

/*
 printToScreen: Method that calls for sensor readings and updates the values on the screen
*/
void printToScreen()
{
 // Method call to read sensors
```

```
 readSensors();

 // Convert measurements to strings
 String diaString = String(diameter);
 String rfString = String(radForce);

 // Print measurements as strings to the screen
 myNex.writeStr("t1.txt", diaString);
 myNex.writeStr("t2.txt", rfString);
}

/*
 readRotEncPos: Method used to read measurements from the rotary encoder
*/
void readRotEncPos()
{
 // Set state
 aState = digitalRead(CLK);

 // Check if state has changed and if the motor is moving
 if (aState != aLastState && motorFlag == true){
     // Increase diameter reading
     if(rotEncFlag == true&&diameter <= 62)
     {
       diameter+=0.79;
     }
     // Decrease diameter reading
     else if(rotEncFlag == false&&diameter>2)
     {
       diameter-=0.79;
       Serial.println(diameter);
     }
    }
   // Save state of rotary encoder
  aLastState = aState;
}

/*
 readFSR: Method used to get the output voltage of the force sensitive resistor
*/
void readFSR()
{
 // Read voltage from the pin the FSR is attached to
 int fsrAnalogPin = 0;
```

```
 int fsrSignal = analogRead(fsrAnalogPin);

 // Map the voltage to a higher value for easier calculations
 int fsrVoltage = map(fsrSignal, 0, 1023, 0, 5000);
 initFSRVolt = fsrVoltage;
}

/*
 readSensors: Method used to take measurements from both sensors
*/
void readSensors()
{
// Calls methods for both individual sensors
readRotEncPos();
readFSR();

// Hard coding the radial force based of of the diameter of the stent
// Method the mechanical engineering team desired for this
if (diameter < 56 && diameter > 39)
{
 radForce = 0;
}
else if(diameter >=62)
{
 radForce = 0;
}
else
{
  radForce = (1.3347*diameter*diameter-126.91*diameter + 2908.9)/1000;
}
}

/*
 rotateCCW: Method used to rotate the motor counter clockwise
*/
void rotateCCW()
{
 // Set appropriate flags
 motorFlag = true;
 rotEncFlag = true;

 // Set direction of motor
 digitalWrite(dir, HIGH);
```

```
 // Rotate motor
 pulse = !pulse;
 digitalWrite(pul, pulse);
}

/*
 rotateCCW: Method used to rotate the motor clockwise
*/
void rotateCW()
{
 // Set appropriate flags
 motorFlag = true;
 rotEncFlag = false;

 // Set direction of motor
 digitalWrite(dir, LOW);

 // Rotate motor
 pulse = !pulse;
 digitalWrite(pul, pulse);
}

/*
 setDiameter: Method used to rotate motor until a set diameter has been reached
 Param: setToValue - diameter the stent should be crimped to
 Returns boolean indicating if process has succeeded
*/
void setDiameter(int setToValue)
{
 // Set flag to decrease or increase diameter respectively
 if (diameter > setToValue)
 {
   diaFlag = true;
 }
 else if(diameter < setToValue)
 {
   diaFlag = false;
 }

 // Call helper function
 setDiaHelper(setToValue);
}

void setDiaHelper(int setToValue)
```

```
{
 // Check if motor should be stopped
 if(stopFlag==true)
 {
   return;
 }

 // Listen for interrupts from the screen
 myNex.NextionListen();

 // Rotate motor in correct direction
 if (diameter > setToValue)
 {
   // Check
   if(diaFlag == true)
   {
   rotateCW();
   readSensors();
   printToScreen();
   setDiaHelper(setToValue);
   }
 }
 else if (diameter < setToValue)
 {
   if(diaFlag == false)
   {
   rotateCCW();
   readSensors();
   printToScreen();
   setDiaHelper(setToValue);
   }
 }
}

/*
 setRadForce: Method used to rotate motor until a radial force has been reached
 Param: setToValue - radial force the stent should be crimped to
 Returns boolean indicating if process has succeeded
*/
void setRadForce(int setToValue)
{
 myNex.NextionListen();

 if(stopFlag==true)
```

```
 {
   return;
 }


 // Measured radial force is less than specified
 if (radForce < setToValue);
 {
   rotateCW();
   readSensors();
   printToScreen();
   setRadForce(setToValue);
 }
}

/*
 setupMotor: method used to set signals sent to stepper motor driver board
*/
void setupMotor()
{
 pinMode(ena, OUTPUT);
 pinMode(dir, OUTPUT);
 pinMode(pul, OUTPUT);
 digitalWrite(ena, HIGH);
 digitalWrite(dir, HIGH);
 digitalWrite(pul, HIGH);
}

void softReset()
{
 digitalWrite(resetPin, LOW);
}

void setup(){
 pinMode(CLK,INPUT_PULLUP);
 pinMode(DT,INPUT_PULLUP);
 digitalWrite(resetPin, HIGH);
 pinMode(resetPin, OUTPUT);
 aLastState = digitalRead(CLK);
 attachInterrupt(digitalPinToInterrupt(2), softReset, FALLING);
 setupMotor();
 myNex.begin(9600);
}
```

```
void loop(){
 stopFlag = false;
 motorFlag = false;
 printToScreen();
 myNex.NextionListen();
}

/*
 Trigger for start button
*/
void trigger0(){
 String inputStr = myNex.readStr("t0.txt");
 int inputNum = inputStr.toInt();

 if(myNex.readNumber("c0.val")==1)
 {
   if(inputNum <= 1000&&inputNum > 0)
   {
     setDiameter(inputNum);
   }
   else
   {
     myNex.writeStr("t3.txt", "Enter value within range");
   }
 }
 else if(myNex.readNumber("c1.val")==1)
 {
   if(true)//inputNum <= 10 && inputNum > 0)
   {
     setRadForce(inputNum);
   }
   else
   {
     myNex.writeStr("t3.txt", "Enter value within range");
   }
 }
}

/*
 Trigger for reset button
*/
void trigger1()
{
 stopFlag = true;
```

```
}
/*
 Trigger for open iris button
*/
void trigger2()
{
 setDiameter(62);
}

void trigger3()
{
 while(initFSRVolt < 1000)
 {
   readFSR();
   delay(60);
   rotateCCW();
   diameter = 62;
 }
}
```