Automatic Blood Pressure Measurement

Lilly Chiavetta

High blood pressure is a condition that affects many people's health. At home blood pressure measurements are helpful for individuals with higher blood pressures to ensure their health. Using a battery powered air-pump system, a portable blood pressure cuff was made. Microphones were used to identify points of turbulent flow within the arteries and a pressure sensor was used to identify the pressure at these points. System control was achieved with an Arduino Nano Every identifying when microphones experienced peaks in noise. The device was tested against criteria for portability and accuracy in different environments. The device was concluded effective in ideal conditions, with an error of about ±1.6mmHg systolic and ±0.8mmHg diastolic.

I. INTRODUCTION

Human blood pressure is a good indication of cardiovascular health and serves as a measure of the force exerted by blood against arterial walls while it is pumped through the body. Maintaining optimal blood pressure is crucial for overall physical well-being because high blood pressure, also known as hypertension, is a significant risk factor for heart disease, stroke, and other serious health conditions [1].

Blood pressure typically consists of two values: systolic and diastolic pressure. Systolic pressure represents the force when the heart contracts, while diastolic pressure measures the pressure in the arteries when the heart is at rest between beats [1]. Blood pressure measurements are done by first constricting a blood pressure cuff around the arm, tightening arteries so no blood can flow through. Once the pressure begins to release, it will fall below systolic pressure, allowing blood to flow through the artery. The release of pressure causes turbulent blood flow and causes vibrations in the arterial walls. These vibrations can be heard through a stethoscope as Korotkoff sounds, which sound like tapping or knocking [2]. As the cuff pressure decreases, the turbulent flow lessens, lessening the amplitude of the Korotkoff noises. Eventually, the flow becomes laminar and silent [2]. The pressure range where Korotkoff noises are heard are considered the individual's systolic and diastolic pressures.

The device built consists of a pair of microphones to measure Korotkoff sounds, a pressure sensor to measure pressure of the cuff, and an Arduino Nano Every to regulate the OLED display and pressure pumps. It is equipped with standard safety regulations such as an automatic release button and will never increase the pressure above 210 mmHg. The specifications for this device were if it can operate in loud environments, can operate on a moving individual, is portable and lightweight, can fit within a surface area of a small adult arm (22cm), and can accurately identify blood pressure systolic and diastolic values within ±5 mmHg. The device was tested against all criteria and compared to a commercial device.

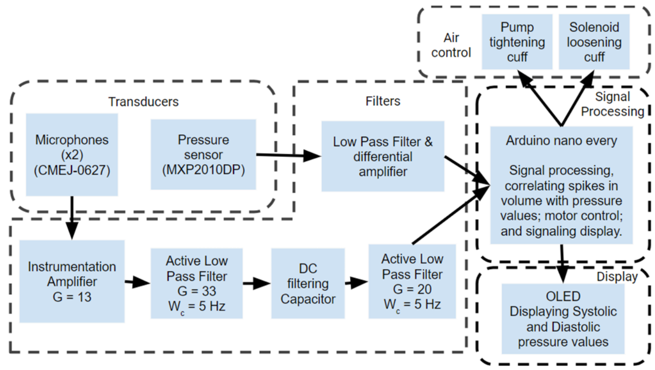

Fig. 1. Block diagram of blood pressure measurement system.

II. METHODS

The design of the device can be broken down into audio transducing and filtering, pressure transducing and filtering, interactive elements, and system control. All these components come together to deliver a fully autonomous blood pressure measurement system as seen in Fig. 1.

A. Audio Transducing and Filtering

Korotkoff sound detection was done by using two microphones, one placed on the brachial artery at the elbow below the cuff, and the other placed on the upper arm above the cuff. Both microphones' initial input is fed into an instrumentation amplifier with a gain of 13. This removes ambient noise so that only sound differences between the two (the only expected sound is the sound made at the artery) are amplified; so loud noises such as music or talking picked up by both microphones will be removed from the audio output.

Once the differential signal is isolated, it is passed through an active low pass filter with a gain of 30 and a cutoff frequency of 5Hz. This is then fed into a DC filter and a second active low pass filter with a gain of 20 and a cutoff frequency of 5Hz. The final output is centered around a virtual ground of 2.5V so that it can be easily input into the Arduino's ADC. The full microphone filtering schematic can be seen in Fig. 2. Microphone accuracy was tested by viewing the output on an oscilloscope while the arm was constricted between systolic and diastolic pressure and viewing if sounds were heard.

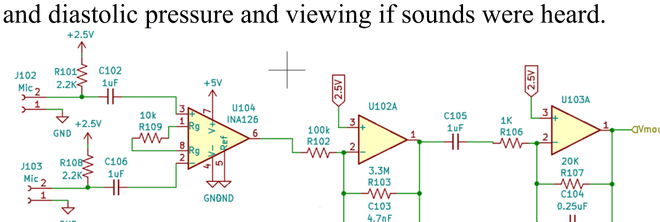

Fig. 2. Microphone input and filtering schematic.

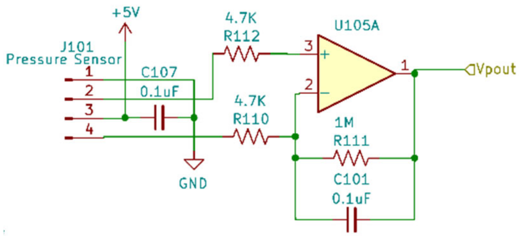

Fig. 3. Pressure input and filtering schematic.

B. *Pressure Transducing and Filtering*

Pressure transducing the cuff pressure was done by a differential pressure sensor connected to the cuff through surgical tubing. The output of the pressure sensor was then run through a differential amplifier and active low pass filter as seen in the circuit schematic in Fig. 3. The differential amplifier induced a gain of $200(V_3 – V_2)$ and lowpass cutoff frequency of 1.5Hz. The output of the system was then calibrated with a linear fit by measuring the voltage at different known pressures at 10mmHg intervals.

C. *Interactive Elements (Display and Pumps)*

To inflate the cuff pressure, a motorized air pump was used. This was powered by an external power circuit with input from a 9V battery connected to a 5V LDO regulator. To control power delivery to the motor, a mechanical relay switched by an Arduino signal was added. This allows for the strain of motor use to not affect Arduino operation as the maximum current from its onboard regulator is 500mA.

Also connected directly to battery power is a solenoid to quickly release the pressure by connecting the air system to ambient air. This is also controlled by a separate relay switched on or off by the Arduino. This switch occurs after a diastolic measurement has been recorded. The device uses passive deflation when measuring systolic and diastolic pressures. If the user encounters a situation where the cuff must immediately deflate, an automatic release button was included to bypass the relay, directly powering the solenoid, and releasing pressure.

Systolic and diastolic result display was done with an OLED. The measured values would be displayed after the system took both measurements with both values highlighted.

D. *System Control*

System control is done by an Arduino Nano Every employing a peak detection algorithm on the incoming filtered microphone output voltage. The system works by first inflating the cuff until it can either no longer hear a heart rate or reaches 210mmHg to prevent dangerous pressures or if a heart rate was not detected during inflation. Once the system is inflated, the Arduino listens for sounds from the microphone. If the sound is above a certain threshold set as 10% greater than the average of the last 50 measured values, it is determined as a peak. The first peak detected is correlated with the current pressure measured by the pressure sensor as the systolic pressure. Once there are no more peaks detected in the last 100 values, the last peak detected is correlated with its associated pressure as the diastolic value. The Arduino will then signal the solenoid to release and display the measured pressure values.

E. *Testing Protocol*

Before testing the device, the circuit was soldered to prevent potential breadboard induced errors and give a full estimate of size. Size and weight were tested by using a ruler to understand the area of the device and a scale to measure weight. Portability was also tested by measuring the expected device lifetime with a 9V battery.

Accuracy and precision tests were done by getting 8 measurements with the system on one individual in ideal conditions (microphone placed on artery, participant is stationary, and no loud noises) and comparing them to each other and a gold standard commercial blood pressure cuff. They were also done in non-ideal conditions with the participant singing to Taylor Swift, dancing, and with the microphone not placed over the artery. ± Error ranges were determined with the interquartile range of measurements in ideal conditions.

III. RESULTS

A. *Calibration*

To validate the linearity of pressure sensor measurements and create an accurate equation to calculate pressure with, the output voltage of the pressure sensor was taken at different pressures. The data was then plotted in Fig. 4 to visualize the linearity of the data. When plotting a line of best fit along the measured pressures, the output voltage was determined to relate to the pressure linearly with $R^2 \approx 1$. The fit equation was determined as:

$$P_{cuff} = 91.3\ \frac{mmHg}{V} V_p - 242.3mmHg$$

Korotkoff noise detection was tested by measuring the microphone output while the arteries underwent turbulent flow (Fig. 5). It was determined that the microphones could successfully detect sounds by visual determination and correlating spikes in volume with heart rate. Time delay was not noticeable, and it was determined that as pressure released, the sounds also died out. The amplitude of the noise was about 50 mV, while the amplitude of the quietest Korotkoff sounds measured by the mic was 150mV. The peak detection algorithm worked 95% of the time, with the loss of peaks normally happening at the end of the systolic-diastolic range where the sound is quieter.

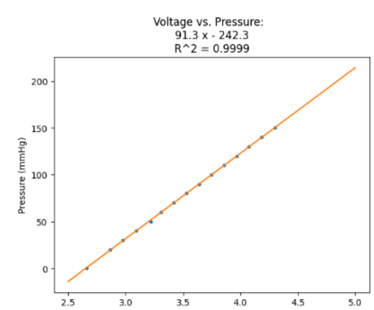

Fig. 4. Pressure sensor linear calibration line. Raw (blue) data points have an extremely linear relationship with each other with an R^2 of about 1.

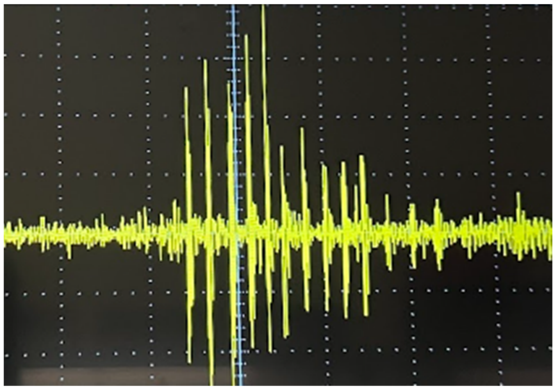

Fig. 5. Oscilloscope reading of filtered and amplified dual microphone output. Spikes in voltage correlate with heartbeats and noise from turbulent flow within the arteries.

B. Portability

The device was measured to be 19.7cm, less than the surface area of most standard blood pressure cuffs, meant to range from arm areas of 20cm to 42cm [3]. The weight was also measured at 0.456kg, in the midrange of commercial options, which weigh around 0.2 to 0.7kg [3]. Motor current consumption is 507mA within the approximately 10 second inflation period. Using a 9V alkaline battery with a life of about 550 mAh will provide power for about 390 uses. Qualitatively, the device appears compact and adaptable to multiple part configurations so that the best arm fit is achieved.

C. Accuracy and Precision in Ideal Conditions

The device was tested 8 times on a single participant in ideal conditions against a commercial device as the gold standard. The true value of the participants blood pressure that day was given as 122/87 mmHg. After completing 8 rounds of measurement on the device, the means of the measured device values were 122.51/86.1mmHg. This had an interquartile range of 3.17 for systolic and 1.6 diastolic, giving an error range of error range of ±1.6 mmHg systolic and ±0.8 mmHg diastolic. The percent error compared to the gold standard was 0.42% for systolic and 1.04% for diastolic.

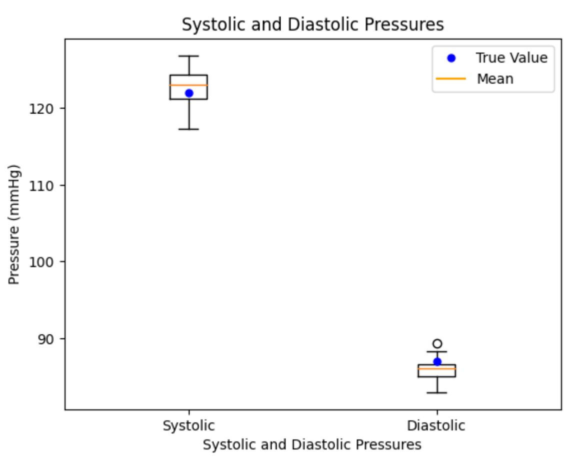

Fig. 6. Box plot of systolic and diastolic values at ideal conditions compared to true values (blue). Systolic measurements have a 3.17mmHg interquartile range and Diastolic measurements have a 1.6mmHg interquartile range.

D. Non-Ideal Conditions

The device was also tested with the microphone moved and with subject moving and singing while getting blood pressure measurements. 3 trials were conducted for each condition.

When moving the top microphone around the upper arm, there were no significant changes in reported values, with a percent error of 1.45% systolic and 0.25% diastolic compared to the true value measured by a commercial device. The interquartile ranges were 4.43 systolic and 2.12 diastolic. However, when moving the lower microphone off the artery, no peaks could be detected by the Arduino over the ambient noise. Therefore, no pressure readings could not be correlated with peaks in turbulent flow and systolic and diastolic values could not be determined.

While signing loudly to Taylor swift when the device was operating, there were no significant changes compared to ideal values. The device was still able to detect peaks in the Korotkoff sounds and had a percent error of 2.05% systolic and 0.94% diastolic compared to the true value measured by a commercial device. The interquartile ranges were 5.02 systolic and 1.84 diastolic. However, if the participant turned to face the device top microphone while singing, peaks would be detected when they were not truly happening since there was a noise difference. This would cause a larger systolic value and smaller diastolic value than expected dependent on the pressure values measured when the participant started singing. If the participant stopped signing or turned away before diastolic pressure was reached, a false diastolic reading could be triggered and result in a pressure value higher than expected. The system would identify the last "peak" as when the participant stopped singing or turned away from the microphone rather than the true diastolic value.

When physically moving by dancing, there were major changes in device function. Most values read were wildly off from their expected with 25.68% error for systolic and 53.26% error for diastolic. Interquartile ranges also were quite high with a range of 11.5 systolic and 23.0 diastolic. There was also a measurement with a diastolic value (measured as 132.33mmHg) reported higher than the systolic value (measured as 127.67mmHg), which is not humanly possible.

IV. DISCUSSION

Overall, the device was proven accurate in ideal conditions with the error range of ±1.6mmHg systolic and ±0.8mmHg diastolic and a percent error of 0.42% systolic and 1.04% diastolic compared to commercial products. This high accuracy is likely due to Korotkoff sounds being the current gold standard for manual measurement of blood pressure and the placement of the microphone directly over the artery.

The pressure sensor chosen was proven to be a very accurate measure of pressure and calibration proved a very close fit to a linear relationship of the pressure measured and voltage output (Fig. 4). This linear relationship gives great confidence in using a linear equation for the chosen pressure sensor's applications, especially with regards to measuring pressure of the cuff.

The use of two microphones proved very effective at removing loud ambient noise when used with the device. The larger percent errors and interquartile ranges found when singing loudly are likely due to a lack of trials since only 3 were used. However, the percent error is still below 5% and the ± error ranges were both below ±5mmHg as desired. Korotkoff sounds were detected with 95% accuracy, proving the peak detection algorithm sufficient for device application.

The portability of the device is highlighted by its compact size, lightweight design, and long lifetime. With a measurement of 19.7cm, the device proves to be smaller than the surface area of most standard blood pressure cuffs, emphasizing its convenience for users with varying arm circumferences. Weighing at 0.456kg, the device falls within the midrange of commercial options, striking a balance between being lightweight and maintaining a substantial build. The motor's current consumption during the 10-second inflation period indicates an efficient energy usage profile. The incorporation of a 9V alkaline battery, often used in smoke detectors, guarantees a long lifetime of 390 uses. Additionally, the qualitative observation of the soldered device being adaptable to most pump layouts reinforces its suitability for diverse user needs. Overall, the combination of size, weight, and adaptability positions the device as a portable and user-friendly solution for blood pressure monitoring, facilitating ease of use in various settings and scenarios.

The device's performance under various conditions reveals important insights into its robustness and limitations. When the top microphone was moved around the upper arm, the device demonstrated reliability, showing negligible changes in reported values and a low percent error compared to a commercial device. However, the challenge arose when the lower microphone was moved off the artery, resulting in an inability to detect peaks in turbulent flow and thus rendering the device incapable of determining systolic and diastolic values.

The most significant challenges emerged during physical movement, specifically dancing, where major deviations from expected values were observed. The substantial errors in systolic and diastolic measurements, along with instances where diastolic values exceeded systolic values, raise concerns about the device's reliability during dynamic activities. These findings emphasize the importance of considering user movements when assessing the device's performance.

With this in mind, it is safe to recommend that this device can be used in medical or at-home applications with a stationary patient with the second microphone above their artery. The low percent error and high precision allows for confidence in its values and its portability allows for it to be used by most adults.

V. CONCLUSIONS

To create an accurate automatic blood pressure system for medical applications, one must consider current systems and the accuracy and precision needed in a medical environment. The device constructed fits these needs substantially. It preforms very similarly to a commercial device It performed well compared to the commercial device with a similar accuracy and them both being negatively affected by movement.

The current device is mainly limited by its necessitated placement over the artery. Research in Korotkoff sound frequency components is limited in the medical community and there does not appear to be a concluded frequency range of Korotkoff sounds. This means that it is possible the low pass filter actually filters out the Korotkoff sound frequency and what the microphone hears is simply just the artery knocking against the transducer.

The device could be improved by exploring alternative batteries with higher capacity to mitigate concerns about prolonged measurement durations, providing a more efficient user experience. Conducting experiments to precisely understand Korotkoff sound frequency composition could unveil valuable insights, guiding refinements to the device's filtering system and potentially overcoming current limitations associated with necessary artery placement. However, the device exhibits promising features and parallels commercial devices in many aspects. It proves itself advanced, reliable, and user-friendly automatic blood pressure monitoring system for medical applications.

REFERENCES

[1] Wilk, B., & Hanus, R. (2019, June). Blood flow in the brachial artery compressed by a cuff. The European Physical Journal Conferences, 213, 02099. https://doi.org/10.1051/epjconf/201921302099

[2] Alvarado Alvarez, M., Padwal, R., & Hiebert, W. (2023, March). Masking of Korotkoff sounds used in blood pressure measurement through auscultation. The Journal of the Acoustical Society of America, 153(3), 1496-1505. https://doi.org/10.1121/10.0017354

[3] Bilo, G., Sala, O., Perego, C., Faini, A., Gao, L., Głuszewska, A., Ochoa, J. E., Pellegrini, D., Lonati, L. M., & Parati, G. (2017). Impact of cuff positioning on blood pressure measurement accuracy: may a specially designed cuff make a difference? Hypertension Research, 40(6), 573-580. https://doi.org/10.1038/hr.2016.184

APPENDIX

A. Soldered Assembly

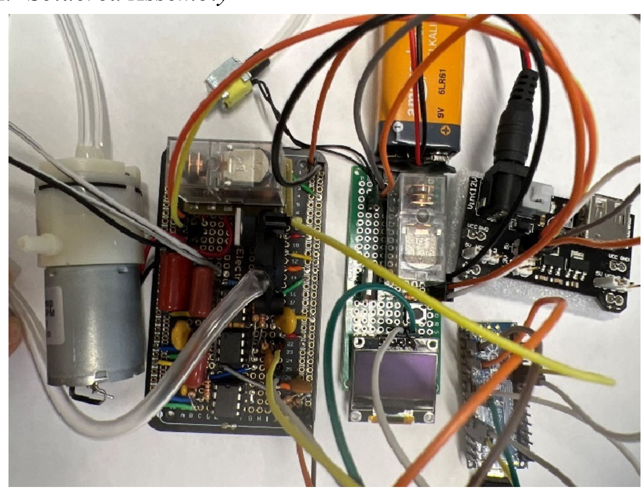

Fig. 7. Soldered assembly including battery, both boards, and Arduino. Wires between boards are long for setup adjustments around blood pressure cuff when fully assembled.

B. Microphone and Cuff Setup:

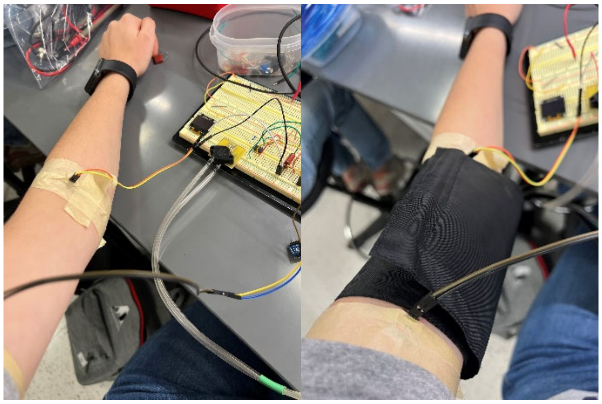

Fig. 8. Microphone setup with and without blood pressure cuff. Microphones placed around the cuff, one over the brachial artery over the elbow, one on the upper bicep.

C. Portability Measures

TABLE I. DEVICE PORTABILITY MEASUREMENTS

Area	Weight	Lifetime
21.7 cm	0.465kg	390 uses

D. Part numbers:

- Microphones: CMEJ-0627
- Pressure sensor: MXP2010DP
- Op Amps: MCP6002
- Air Pump: ZR370-02PM
- Solenoid: Fa0520E
- OLED: UG-2864HSWEG01

E. Arduino code:

```
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "EveryTimerB.h"
#define Timer1 TimerB2 // use TimerB2 as a drop in replacement for Timer1

#define OLED_RESET     -1 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C ///< See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32


#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

//Defining Pins:
#define MIC_PIN A0 //Pin for microphone
#define PRESSURE_PIN A1 //Pin for pressure
#define PUMP_PIN 2 //Pin for pump control
#define RELEASE_PIN 3 //Pin for solenoid control

String inString = "";  // string to hold input- This is for auto-adjusting the offset parameter
int VMAX = 5;
float current_mic_value = 0;
float current_pressure_value = 0;

float k_threshold = 2.9; //threshold of Korotkoff sound voltage transduction to trigger a systolic or diastolic pressure reading
int count = 0;
bool hit_systolic = false;
bool hit_diastolic = false;
float systolic_reading = 0;
float diastolic_reading = 0;

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

void setup() {
 pinMode(PUMP_PIN, OUTPUT);
 pinMode(RELEASE_PIN, OUTPUT);

 Serial.begin(9600); // Serial connection to print samples
 while ( !Serial ) delay(10); //Serial required to start

 if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
   for (;;); // Don't proceed, loop forever
 }

 // Show initial display buffer contents on the screen --
 // the library initializes this with an Adafruit splash screen.
 display.display();
 delay(500);

 // Clear the buffer
 display.clearDisplay();
 drawchar();

 Serial.print("Volume ");
 Serial.println("Pressure");

 delay(500);
 Serial.println("1..");
 delay(500);
 Serial.println("2..");
 delay(500);
 Serial.println("3..");
 delay(500);
 Serial.println("GO!");

 while (threshold > get_pressure(current_pressure_value) && hearheartbeat()) {
   digitalWrite(PUMP_PIN, HIGH);
   current_mic_value = get_raw_value(MIC_PIN); //Getting raw value
```

```
    current_pressure_value = get_raw_value(PRESSURE_PIN); //Getting raw value
    Serial.print(current_mic_value);
    Serial.print(" ");
//    Serial.print(get_pressure(current_pressure_value));
    Serial.print(" ");
    Serial.println(current_pressure_value);
  }
  digitalWrite(PUMP_PIN, LOW);
  Serial.print("Finished pumping at ");
  Serial.print(get_pressure(current_pressure_value));
  Serial.println("mmHg");
}

void loop() {
  while (Serial.available() > 0) {
    byte inChar = Serial.read();
    if (isDigit(inChar)) {
      // convert the incoming byte to a char and add it to the string:
      inString += (char)inChar;
    }
    // if you get a newline, print the string, then the string's value:
    if (inChar == '\n') {
      int value = inString.toInt();
      k_threshold = value / 10.0;
      delay(10);
      inString = "";
      Serial.print("Current pressure: ");
      Serial.println(get_pressure(current_pressure_value));
    }
  }

  current_mic_value = get_raw_value(MIC_PIN); //Getting raw microphone value
  current_pressure_value = get_raw_value(PRESSURE_PIN); //Getting raw pressure value
  Serial.print(current_mic_value);
  Serial.print(" ");
  Serial.print(current_pressure_value);

  if (current_mic_value > k_threshold && !hit_systolic) { //if a KS is heard for the first time
    systolic_reading = get_pressure(current_pressure_value); //calculating pressure
    hit_systolic = true;
    Serial.print(" ");
    Serial.print(1);
  } else if (hit_systolic) {
    if (count > 100) { //If silence is reached
      if (!hit_diastolic) { //If we have not hit diastolic pressure yet
        diastolic_reading = get_pressure(current_pressure_value);//calculating pressure
        hit_diastolic = true;
      }
      Serial.print(" ");
      Serial.print(0);
      digitalWrite(RELEASE_PIN, HIGH);//Release pressure
      drawchar(); //display results
    } else { //If we are inbetween heartbeats
      count ++;
      Serial.print(" ");
      Serial.print(1);
    }
    if (current_mic_value > k_threshold) { //if KS is detected
      count = 0;
    }
  } else {
    Serial.print(" ");
    Serial.print(0); //Systolic pressure not reached and no KS
  }
  Serial.println("");
}

float get_raw_value(int pin_num) {
  //converting the ADC read to a voltage
  float analogDigitalConversion = analogRead(pin_num);
  float voltage = (analogDigitalConversion * VMAX) / (pow(2, 10) - 1);
  return voltage;
}

float get_pressure (float voltage) {
  return 91.3 * voltage - 242.3; //mmHg
}

void drawchar() {
  display.clearDisplay();

  display.setTextSize(1);             // Normal 1:1 pixel scale
  display.setTextColor(SSD1306_WHITE);        // Draw white text
  display.setCursor(0, 0);            // Start at top-left corner
  display.print(F("Systolic: "));
  display.setTextSize(1);             // 2x pixel scale
  display.println(systolic_reading); // display current temp in large font

  display.setTextSize(1);             // Normal 1:1 pixel scale
  display.print(F("Diastolic: "));
  display.setTextSize(1);             // 2x pixel scale
  display.println(diastolic_reading); // display current temp in large font

  display.display();
}
```