Utveckling av en smart spirometer för mätning av lungfunktion

Vidareutveckling av mätinstrument med Arduino och sensorer

Examensarbete inom högskoleingenjörsprogrammet Mekatronik och Elektroteknik

Karam Kattah
Ahmad Jouda

INSTITUTIONEN FÖR DATA- OCH INFORMATIONSTEKNIK
MEKATRONIK OCH ELEKTROTEKNIK

CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2026
www.chalmers.se

Förord

Det här examensarbetet genomfördes under vårterminen 2026 inom programmet för Elektroteknik och Mekatronik vid Chalmers tekniska högskola. Arbetet behandlar utvecklingen av en Arduino-baserad prototyp som demonstrerar principen bakom lungfunktionsmätning. Projektet har varit både lärorikt och utmanande. Under arbetets gång har vi praktiskt arbetat med elektronik, sensorer, programmering, signalbehandling och felsökning. Lärandet har till stor del byggt på att prova olika lösningar, identifiera problem och steg för steg förbättra systemet. Prototypen är inte framtagen för att vara ett medicinskt godkändAI-verktyg utrustning, utan ett tekniskt demonstrationssystem för utbildning och undervisning. Användningen av AI verktyg har endast varit begränsad till språkgranskning, textstruktur och figurvisualisering. De tekniska lösningarna, kopplingarna, programmeringen och slutsatserna har utformats och kontrollerats av oss själva. Vi vill rikta ett stort tack till vår handledare Sakib Sistek för hans engagemang, vägledning och sitt stöd under hela projektets gång. Vi vill också tacka Ants Silberberg, universitetsektor inom medicinteknik, som tog sig tid att diskutera projektet och ge feedback kring mätprinciper och sensorteknik. Slutligen vill vi också tacka Chalmers för de resurser och den inspirerande miljön som har gjort projektet möjligt.

Sammanfattning

Spirometri är en vanlig metod för att undersöka lungfunktion och används bland annat för att mäta hur mycket luft en person kan blåsa ut och hur snabbt utandningen sker. Kommersiella spirometrar kostar ofta 4 000–25 000 kr och är anpassade för klinisk miljö. Det här projektet undersökte om samma principer kan demonstreras för ungefär 1 300 kr med lättillgängliga komponenter.

Prototypen är baserad på en Arduino Uno, en differentialtrycksensor MPX5010, OLED-display, SD-kortmodul, Bluetooth-modul, knappar, LED, buzzer samt en analog filter och förstärkarkrets med operationsförstärkaren LM358. Tryckförändringar genereras och testas med en slang och sprutuppställning som efterliknar en Venturi-liknande mätning. Sensorn omvandlar tryckförändringen till analog signal som filtreras, förstärks och läses av Arduino.

Hårdvara och mjukvara har utvecklats för att demonstrera en mätkedja från tryck-signal till uppskattade lungfunktionsvärden. Flera delsystem testades separat. Vid sluttestningen kunde systemet inte verifieras som stabilt i sin helhet. Resultatet blev därför en demonstrationsprototyp som visar principen bakom en enkel digital spirometer, men inte ett färdigkalibrerat eller medicinskt verifierat mätsystem.

Terminologi och förkortningar

Tabellen nedan visar begrepp och förkortningar som används i projektet.

Tabell 1

Term / förkortning	Förklaring
Arduino	Den centrala styrenheten i systemet
MPX5010	Differentialtrycksensor från NXP, mätområde 0–10 kPa
Tryck (pressure)	Skillnad i lufttryck mellan två punkter, mäts i Pa
Luftflöde (flow)	Volym luft per tidsenhet, uppskattad ur trycket
FVC	Total forcerad utandningsvolym
FEV1	Utandad volym efter 1 s
OLED-display	Skärm som visar mätdata
I2C	Seriell buss för displaykommunikation
SPI	Seriell buss för SD-kort
Bluetooth (HM 10)	Trådlös överföring
SD-kortmodul	Läser och skriver microSD-kort via SPI
LED	Visuell återkoppling
Buzzer	Hörbar återkoppling till användaren
Δt	Tidsintervall mellan mätningar
Progressbar	Grafisk indikator för hur stor del av mätningen som återstår
Graf (flödesgraf)	Realtidskurva över luftflöde under mätningen
Analog signal	Kontinuerlig signal från sensorn
Digital signal	Diskret 0/1-signal för knappar, LED och summer
Debounce	Metod för att filtrera knapptryck
Bandpassfilter	Filter som stabiliserar signalen
Operationsförstärkare (Op amp)	Förstärker och filtrerar signaler
Kondensator	Används i filter
Resistorer	Begränsar ström och används i filter
VCC	Komponentspänning
GND	Nollpotential
A0	Analog mätingång
ADC	A/D omvandlare
SDA	Seriell datalinje
SCL	12C klocklinje
MOSI	Skicklinje (SPI)
MISO	Mottaglinje (SPI)
SCK	SPI klocklinje
CS	Enhetsvalssignal
FAT32	SD-kortets filformat
TXD	TX pin
RXD	RX pin
Digital filtrering	Mjukvarufiltrering
Seriell kommunikation	Bitvis dataöverföring
BLE Serial	Bluettoth test app

Innehåll

1 Inledning

Teknikens betydelse inom medicin och hälsovård ökar, särskilt när det gäller att mäta och följa kroppens funktioner. Med hjälp av sensorer, mikrokontroller och digitala visualiseringar kan fysiologiska signaler registreras på ett kostnadseffektivt sätt. I detta projekt undersöks hur grundläggande elektroniska komponenter kan användas för att demonstrera principen bakom spirometri. Spirometri är en metod för att mäta lungkapacitet, till exempel hur mycket luft en person kan blåsa ut och hur snabbt utandningen sker. Genom att koppla en trycksensor till en Arduino kan tryckförändringar vid utandning registreras, bearbetas och visas i realtid. Detta är tillräckligt för att illustrera de grundläggande principerna bakom spirometri, vilket är projektets syfte.

1.1 Problemformulering

De spirometrar som används inom sjukvården är anpassade för klinisk diagnostik och är ofta dyra. Därför passar de inte alltid för undervisning eller enklare demonstrationer. I detta projekt undersöktes möjligheten att bygga en billigare spirometerprototyp med lättillgängliga komponenter. Arbetet fokuserade på att mäta tryckförändringar vid utandning, behandla sensorsignalen och presentera uppskattade lungfunktionsvärden på ett tydligt sätt. Den huvudsakliga frågan var om principen bakom spirometri kan demonstreras med en Arduino, en differentialtrycksensor och enkel signalbehandling, utan att systemet behöver vara medicinskt godkänd utrustning.

1.2 Mål och avgränsningar

Målet var att utveckla en Arduino-baserad spirometerprototyp som mäter tryckförändringar, uppskattar luftflöde och visar mätdata på en OLED-display. Systemet omfattar även funktioner för användarstyrning, återkoppling, datalagring och trådlös överföring. Projektet är begränsat till en teknisk prototyp avsedd för demonstration och undervisning. Systemet är varken medicinskt godkänt eller kalibrerat mot en klinisk spirometer och ska därför inte användas för diagnostik eller självbedömning. De uppmätta värdena för tryck, luftflöde, volym, FVC och FEV1 utgör tekniska uppskattningar. Spruta och slang användes som luftkanal i prototypen.

2 Teori

För att skapa förståelse för prototypens funktion här de viktigaste begreppen och tekniska principerna som projektet grundar sig på, med särskilt fokus på spirometri, sensorteknik och enkel mätdatahantering.

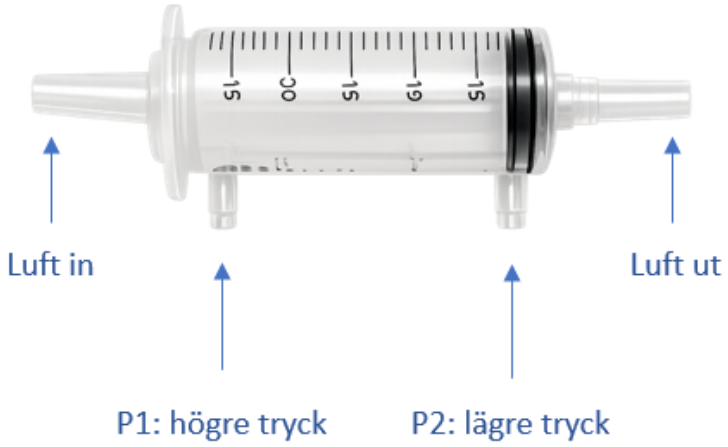
2.1 Lungfunktion och spirometri

Lungorna huvudsakliga uppgifter är att tillföra syre till blodet och att göra sig av med koldioxid. Hur effektivt detta sker påverkas bland annat av hur stor volym lungorna har och hur snabbt luften kan passera in och ut genom luftvägarna. Om lungfunktionen är nedsatt kan det yttra sig som minskad volym, begränsat luftflöde eller båda. Spirometri är en vanlig metod för att mäta lungfunktion. Vid undersökningen får individen först andas in maximalt och sedan blåsa ut så snabbt och kraftfullt som möjligt genom ett munstycke. Under utandningen registreras förändringar i volym och flöde över tid.

Resultat från spirometriundersökningar presenteras ofta genom två viktiga parametrar: FVC (Forced Vital Capacity) och FEV1 (Forced Expiratory Volume in 1 second). FVC beskriver den totala mängden luft som kan blåsas ut efter en maximal inandning, medan FEV1 anger hur mycket luft som andas ut under den första sekunden. Kvoten mellan FEV1 och FVC används inom vården för att skilja mellan obstruktiva sjukdomar, som astma och KOL, och restriktiva tillstånd där lungvolymen är minskad [3]. Utifrån dessa mätprinciper syftar projektet till att återskapa spirometrins metodik och uppskatta värden för FVC och FEV1, snarare än att uppnå samma precision som kliniska instrument.

2.2 Sensorer och mätprincip

Projektet använder en MPX5010 trycksensor för att registrera tryckvariationer under utandning. Sensorn omvandlar tryck till en elektrisk signal som Arduino tolkar via en analog ingång. När luft flödar genom systemet uppstår en tryckförändring som används för att uppskatta luftflödet. Figuren nedan visar hur luftflöde genom sprutan skapar en tryckskillnad mellan två mätpunkter p1 och p2.



Figur 2.2: Venturi-liknande luftkanal med två mätpunkter.

MPX5010 valdes eftersom sensorn kan mäta differentialtryck mellan 0 och 10 kPa och ger en analog utsignal som Arduino kan läsa av. Det gör den passande för en enkel demonstrationsprototyp där små tryckförändringar i luftkanalen ska registreras. Sensorn mäter inte luftflödet direkt, utan tryckskillnaden används för att uppskatta det.

Den här metoden är enklare än den som används i medicinsk utrustning, men den är tillräcklig för att visa hur sensordata kan användas för att analysera luftflöde i en teknisk prototyp. Signalen används även för att uppskatta volym, FVC och FEV1. Tidigare studier visar att trycksensorer tillsammans med mikrokontroller kan användas för att uppskatta luftflöde och beräkna lungfunktionsvärden [3].

2.3 Mikrokontroller och datainsamling

Arduino är systemets styrenhet och hanterar inläsning av sensordata, signalbehandling och styrning av övriga komponenter. Trycksensorn är kopplad till den analoga ingången A0, vilket gör att programmet kan läsa spänningsvariationer från sensorn. Programmet körs i en loop där nya mätvärden samlas in, omvandlas och visas på displayen.

Arduino kontrollerar även knapptryckningar för start och återställning samt aktiverar summer och lysdiod när det behövs. Programmet håller också koll på tiden. Tiden används för att avgränsa mätperioden, registrera FEV1 efter den första sekunden och integrera flödet till volym, se avsnitt 2.6.

2.4 Datavisualisering

I den vidareutvecklade versionen av systemet har displayen kompletterats med en graf som visar luftflödet över tid, vilket gör det lättare att följa mätprocessen i realtid. Dessutom har en progressbar integrerats för att tydligt visa hur långt mätningen har kommit. Displayen växlar mellan två lägen: under pågående mätning visas grafen, progressbaren och direkt återkoppling, medan det slutliga resultatet presenteras när mätningen är avslutad.

2.5 Tidigare tekniska lösningar och forskning

Tidigare forskning har visat att det är möjligt att bygga enklare spirometrar med hjälp av en trycksensor och en mikrokontroller. Principen bygger ofta på att luft passerar genom ett rör med varierande tvärsnitt, till exempel ett venturirör. När luften passerar genom detta rör uppstår en tryckskillnad mellan två mätpunkter, vilket kan användas för att uppskatta luftflödet [1].

Ett exempel på ett sådant system presenteras i [3], där en MPX5500DP sensor kopplas ihop med en Arduino Uno för att registrera utandningsflöden och beräkna lungfunktion. Studien visar att MPX serien kan känna av små tryckförändringar och ge en signal som Arduino lätt kan läsa av. Liknande resultat finns i [4], där en venturibaserad spirometer testas för att övervaka luftvägar hemma. Det finns också nyare lösningar där sensorerna kopplas trådlöst till en mobil. I [2]presenteras en spirometer som skickar mätdata direkt till en app i mobilen. Det har varit en viktig inspiration för Bluetooth-delen i vårt system.

2.6 Förenklad beräkningsmodell: tryck, flöde och volym

Projektet tillämpar en förenklad beräkningsmodell där trycksensorns analoga signal omvandlas stegvis till uppskattade lungfunktionsvärden. Modellen delas in i tre huvudsteg: signal till tryck, tryck till flöde och flöde till volym.

Steg 1: Signal till tryck
MPX5010 ger en analog spänningssignal som varierar med tryckskillnaden mellan sensorns två portar. Arduino läser signalen via den analoga ingången A0. Signalen omvandlas till ett digitalt ADC-värde och används därefter för att uppskatta trycket i pascal.

Steg 2: tryck till flöde
När luft passerar genom luftkanalen uppstår en tryckskillnad mellan två mätpunkter. Modellen utgår från att en större tryckskillnad motsvarar ett högre luftflöde. Luftflödet mäts alltså inte direkt, utan uppskattas utifrån tryckskillnaden. Eftersom luftkanalen inte är kalibrerad mot en spirometer ska flödet ses som ett uppskattat värde.

Steg 3: Flöde till volym
Volymen uppskattas genom att summera flödet över tid. Vid varje avläsning multipliceras det uppskattade flödet med tidsintervallet t mellan två sensoravläsningar:

$$\Delta V = Q \cdot \Delta t$$

Där ΔV är ett litet volymbidrag, Q ett uppskattat luftflöde och Δt tidsintervallet mellan två avläsningar. Den totala volymen beräknas genom att summera alla volymbidrag:

$$V = \sum Q \cdot \Delta t$$

FVC motsvarar den uppskattade totala volymen under mätperioden. FEV1 motsvarar den uppskattade volymen under den första sekunden. Modellen bygger på flera förenklingar, bland annat att tryckskillnaden kan kopplas till luftflödet, att luftkanalen är tillräckligt tät och att tidssteget Δt hanteras stabilt. Eftersom systemet inte är kalibrerat ska de beräknade värdena ses som tekniska uppskattningar och inte som kliniska mätvärden.

3 Metod

Projektet utfördes systematiskt för att prototypen på ett kontrollerat sätt skulle kunna utvecklas, testas och felsökas. I detta kapitel beskrivs arbetsgången, från planering till funktionstest.

3.1 Arbetsmetod och strategi

Projektet genomfördes stegvis, där varje komponent testades separat innan den kopplades in i det övriga systemet. Syftet med detta arbetssätt var att upptäcka och åtgärda fel tidigt, i stället för att behöva felsöka ett större sammankopplat system i efterhand. Arbetet inleddes med att projektets mål och avgränsningar fastställdes, följt av en kort litteraturgenomgång av liknande Arduino-baserade spirometerprojekt.

Parallellt med litteraturstudien påbörjades hårdvaruutvecklingen. Prototypen byggdes upp successivt, från ett grundläggande sensorsystem till en mer komplett lösning med display, knappar, SD kort och Bluetooth. Genom att testa en komponent i taget kunde eventuella problem identifieras och lösas innan nästa del integrerades i systemet.

När problem uppstod felsöktes de direkt innan arbetet fortsatte. Vid några tillfällen behövde tidigare lösningar ändras, men överlag fungerade det stegvisa arbetssättet väl. Slang och spruta användes som en enkel luftkanal för att skapa tryckförändringar.Vid sluttestningen kunde systemet inte verifieras fullt ut som en komplett enhet.

3.2 Marknadsstudie och behovsundersökning

Inför projektet gjordes en enklare marknadsstudie för att se vilka spirometerlösningar som redan finns och om det finns behov av en billigare prototyp. Det framkom att kommersiella spirometrar ofta har hög noggrannhet och flera avancerade funktioner, men att de främst är anpassade för klinisk diagnostik.

De kan dessutom vara dyra, ofta över 1 200 dollar, vilket gör dem mindre lämpliga för undervisning och enklare tekniska demonstrationer. Därför fanns det ett intresse av att undersöka om en billigare lösning kunde byggas med mer lättillgängliga komponenter. Tidigare studier visar att Arduino-baserade system med trycksensorer kan användas för att ta fram enklare spirometerprototyper till en betydligt lägre kostnad än kommersiella system [2][4]. Detta blev en viktig grund för valet av komponenter och systemdesign i projektet.

3.3 Konstruktion och prototyputveckling

Prototypen utvecklades i flera steg. Det första momentet var att ansluta trycksensorn MPX5010 för att undersöka om sensorn kunde registrera tryckförändringar och ge användbara signaler till systemet. Sensorns VCC och GND kopplades till 5 V respektive GND. Sensorns utgång kopplades först till den analoga filter- och förstärkarkretsen. Därefter anslöts den bearbetade signalen vidare till Arduinos analoga ingång A0.

Komponenterna valdes utifrån projektets mål att bygga en enkel och billig demonstrationsprototyp. Arduino Uno valdes eftersom den är lätt att programmera, har analoga ingångar och fungerar bra för utbildningsprojekt. MPX5010 valdes eftersom den kan mäta differentialtryck och ge en analog signal som Arduino kan läsa av. OLED-displayen valdes för sin förmåga att visa både text och grafik. SD-kortmodulen och Bluetooth HM-10 valdes för att demonstrera datalagring och trådlös överföring [3].

Efter inkoppling testades sensorn via Serial Monitor i Arduino . På detta sätt kunde det säkerställas att sensorn reagerade på tryckförändringar. Under de tidiga testerna varierade mätvärdena ibland även utan tydlig blåsning. Detta kan orsakas av sensorns känslighet och av att slang- och sprutuppställningen inte var helt stabil i början.

Efter att sensorn kopplades in anslöts OLED-displayen för att visa mätdata direkt i systemet. Displayens VCC och GND kopplades till 5 V respektive GND, medan SDA och SCL anslöts till Arduinos I2C-pinnar A4 och A5. Med hjälp av Adafruit biblioteken kunde tryckvärden och flödesvärden visas i realtid på skärmen. Genom justeringar i programkoden förbättrades stabiliteten och minskade displayens flimmer.

I den första versionen visades mätvärdena enbart i sifferform, såsom tryck och uppskattat luftflöde. Detta gjorde det svårt att följa förändringarna i utandningen under mätningen. Därför vidareutvecklades displayen med en realtidsgraf och en progressbar som visar återstående mättid.

Under utvecklingen uppstod ofta problem med instabila sensorvärden. Och för att dämpa störningar användes både analog signalbehandling med filter- och förstärkarkretsar och digital filtrering i programvaran.

Digital filtrering implementerades i programvaran efter att signalen hade lästs in via Arduinos A0-ingång. Syftet var att jämna ut signalen och reducera de små störningar som återstod efter den analoga filtreringen.

Den digitala filtreringen fungerade därför som ett extra steg för att stabilisera mätvärdena. I denna prototyp användes en enkel digital filtermetod, men en mer exakt filteralgoritm bör dokumenteras och testas i framtida arbete. Nästa steg var att lägga till knappar för att styra systemet. START-knappen anslöts till pin D4 och RESET-knappen till pin D5, båda med ett 10 kΩ motstånd för att säkerställa stabila signaler. I programvaran användes debounce för att förhindra att ett knapptryck registrerades flera gånger.

Därefter integrerades en buzzer och en LED som återkopplingssystem. Buzzern kopplades till D8 och LED kopplades till D7 via ett 220 Ω motstånd.Systemet programmerades så att ett kort pip ljuder vid mätstart, LED och buzzer varnar om luftflödet är för lågt, och två korta pip indikerar att mätningen är klar.

När grundsystemet fungerade stabilt i testerna, vidareutvecklades mjukvaran för att kunna uppskatta FVC och FEV1. FVC uppskattas genom att summera det uppskattade flödet under hela mätperioden, medan FEV1 motsvarar den uppskattade volymen efter 1 s. Displayen fick dessutom två olika visningslägen: ett under själva mätningen och ett efter att mätningen är avslutad.

I ett senare steg kopplades en SD-kortmodul in för att möjliggöra lagring av mätdata. Modulen kommunicerar via SPI och ansluts med CS till D10, MOSI till D11, SCK till D13 samt MISO till D12. SD-kortet formaterades till FAT32 innan testningen. Systemet är utformat för att automatiskt spara FVC- och FEV1-värden som en textfil på SD-kortet efter avslutad mätning.

Bluetooth-modulen HM-10 var den sista komponenten som anslöts för att utföra trådlös dataöverföring. Modulen kopplades till Arduino med hjälp av seriell kommunikation via TXD och RXD, som hanterar dataöverföringen. Eftersom Arduino använder 5 V-logik och HM-10 kräver 3,3 V, installerades en spänningsdelare för att skydda modulen.

Arduino skickar mätvärden i textformat via en seriell anslutning till HM-10, som sedan överför informationen trådlöst via Bluetooth till mottagaren. Under projektet användes applikationen BLE Serial för att testa överföringen. En egen mobilapplikation planerades, men utvecklingen kunde inte slutföras inom projektets tidsram.

3.4 Datainsamling och analys

Datainsamlingen baserades på den förenklade beräkningsmodellen som beskrivs i avsnitt 2.6. Arduino läser signalen från MPX5010 via A0, omvandlar den till ett uppskattat tryckvärde och använder detta för att beräkna både flöde och volym över tid. Vid varje avläsning multipliceras flödet med tidsintervallet Δt mellan två mätpunkter och adderas till den totala volymen. I prototypen används ett förenklat Δt baserat på hur snabbt programmet genomför mätcykeln. Om looptiden varierar förändras även Δt, vilket påverkar volymberäkningen och därmed FVC och FEV1. För högre noggrannhet bör programvaran mäta den faktiska tiden mellan avläsningarna, till exempel med millis() eller micros(). Eftersom systemet inte är kalibrerat ska värdena ses som uppskattningar.

3.5 Systemintegration och funktionstest

När alla delar hade kopplats samman gjordes en grundläggande funktionskontroll av systemets elektronik och programvara. Målet var att se till att delarna samarbetade och att det inte fanns några tydliga fel i kopplingarna eller i programmet. Under funktionskontrollen kontrollerades att Arduino kunde kommunicera med OLED displayen. Start och resetknapparna testades också för att se att de kunde styra systemet på rätt sätt. Dessutom kontrollerades att LED och buzzer aktiverades enligt den programmerade logiken.

Den analoga signalvägen från MPX5010 via filter och förstärkarkretsen till Arduinos analoga ingång A0 kontrollerades även på kopplingsnivå. Vid integration av SD kortmodulen uppstod inledningsvis felmeddelandet “SD FEL” på displayen. Felet kunde kopplas till SPI anslutningarna och filsystemet på microSD-kortet. Efter felsökning kontrollerades att CS, MOSI, MISO och SCK var korrekt anslutna och att kortet var formaterat som FAT32.

Vid sammankopplingen uppstod återkommande kontaktfel och instabila signaler. Dessa problem kunde vid flera tillfällen lösas genom att felsöka kablar, breadboardanslutningar och kopplingar mellan komponenterna. Flera delsystem fungerade under tidigare testning, men vid den sista testningen fungerade inte hela systemet stabilt som en komplett enhet. Den troliga orsaken var kontaktproblem i kablarna eller på breadboarden. Därför kunde systemet inte verifieras fullt ut som ett färdigfungerande mätsystem.

4 Design

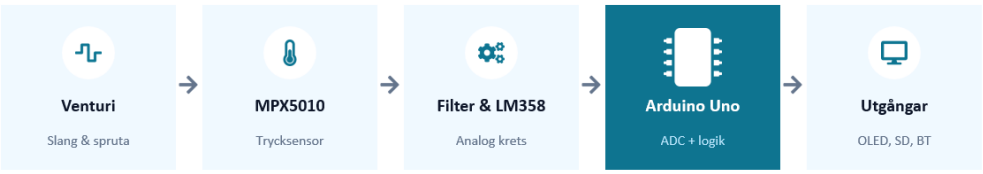
Prototypens design syftade till att integrera mätning, signalbehandling, databehandling och återkoppling. Kapitlet beskriver systemets uppbyggnad och hur de olika delarna samverkar.

4.1 Systemarkitektur

Spirometerns systemarkitektur består av en luftkanal, en trycksensor, analog signalbehandling, Arduino-baserad databehandling samt moduler för visning, lagring och trådlös överföring. Tillsammans bildar delarna en mätkedja där tryckförändringar skapas, registreras, behandlas och bearbetas av Arduino.

Mätningen utgår från en Venturi liknande uppställning där luft leds genom systemet via slang och spruta. När luftflödet förändras uppstår en tryckskillnad mellan två punkter. Denna tryckskillnad överförs till MPX5010 sensorn via slangar och omvandlas till en analog spänningssignal.

Sensorsignalen leds därefter genom en analog signalbehandlingskrets innan den når Arduino. Kretsen består av ett bandpassfilter och två förstärkarsteg med operationsförstärkaren LM358. Syftet är att minska störningar och förstärka den användbara delen av signalen. Den behandlade signalen kopplas sedan till Arduinos analoga ingång A0. Figuren nedan visar mätkedjan från utblåsning till resultat.



Figur 4.1: Mätkedja från utblåsning till resultat.

Arduino är systemets hjärna. Den läser in signalen via A0 och använder programvaran för att uppskatta tryck, luftflöde, FVC och FEV1. Under mätningen uppdateras OLED displayen via I2C med aktuella värden, en flödesgraf och en progressbar. Efter avslutad mätning är systemet avsett att visa resultatvärdena på displayen.

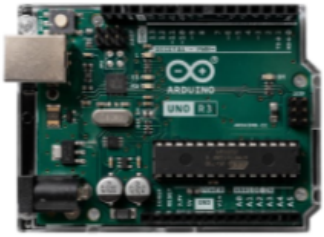
Systemet innehåller också knappar, LED och buzzer för styrning och återkoppling. Startknappen används för att påbörja en mätning och resetknappen för att återställa systemet. LED och buzzer ger visuell och hörbar återkoppling under mätningen.

Mätdata är avsedda att kunna sparas på ett SD-kort och skickas trådlöst via Bluetooth-modulen HM-10 till en enhet. Bluetooth överföringen testades med BLE Serial appen, och en egen app planerades för framtiden.

4.2 Hårdvarudesign

Hårdvaran består av flera komponenter som tillsammans utgör grunden för prototypens funktion. Tillsammans utgörs komponenterna mätning, styrning och överföring av data. I det kommande avsnittet beskrivs varje komponent och dess specifika roll i systemet.

4.2.1 Arduino Uno



Figur 4.2: Arduino Uno som används som mikrokontroller i systemet.

Arduino Uno är projektets huvudmikrokontroller och ansvarar för att läsa av sensordata, bearbeta mätvärdena samt skicka informationen vidare för visning. Kortet är baserat på ATmega328P processorn och programmeras via Arduino IDE i språket C/C++.

4.2.2 Trycksensor MPX5010

Figur 4.3: Trycksensorn MPX5010 som används för att mäta tryckförändringar.

MPX5010 är en analog trycksensor från NXP Semiconductors. Den mäter differentialtryck mellan två portar (P1 och P2) i intervallet 0–10 kPa och ger en analog utgångsspänning som är proportionell mot det uppmätta trycket. I detta projekt används sensorn för att mäta tryckförändringar som uppstår när luft pressas genom systemet, exempelvis vid testning med en spruta. Den analoga signalen läses av Arduinos analog till digital omvandlare (ADC) och omvandlas till ett digitalt värde för vidare uppskattning av luftflöde, FVC och FEV1 i programvaran.

4.2.3 OLED-display

Figur 4.4: OLED-display som används för att visa mätvärden.

En OLED-display används för att visa mätdata i realtid och är kopplad till Arduino via I2C. Under mätningen visas tryck, luftflöde, en graf och en förloppsindikator. Efter mätningen visas resultaten som FVC, FEV1 och maximalt luftflöde.

4.2.4 Start och resetknapp

Figur 4.5: Start- och resetknappar som används för att styra systemet.

Systemet styrs med två knappar: en startknapp och en resetknapp. När startknappen trycks ned påbörjas en ny mätning. Resetknappen återställer systemet till utgångsläget. Båda knapparna är kopplade till digitala ingångar på Arduino och har pull down motstånd på 10 kΩ för att ge stabila signaler. Programmet innehåller dessutom en debounce-funktion som hindrar att ett enskilt tryck registreras flera gånger.

4.2.5 LED

Figur 4.6: LED som används för visuell återkoppling i systemet.

Lysdioden används som visuell återkoppling till användaren. Den är ansluten till en digital utgång via ett strömbegränsande motstånd på 220 Ω och tänds när luftflödet är otillräckligt eller när systemet på annat sätt vill signalera ett tillstånd till användaren.

4.2.6 Buzzer

Figur 4.7: Buzzer som används för hörbar återkoppling i systemet.

Summern används för hörbar återkoppling i tre situationer. Den ger en kort signal när mätningen startar, upprepade signaler om luftflödet är för lågt, och två korta signaler när mätningen avslutas. Det gör systemet enklare att förstå och använda utan att användaren behöver titta på displayen.

4.2.7 Motstånd

Motstånd används i systemet för att skydda komponenter, begränsa ström och skapa stabila signalnivåer. Ett 220 Ω motstånd används tillsammans med lysdioden för att begränsa strömmen. Vid tryckknapparna används 10 kΩ motstånd som pull down motstånd. Detta gör att insignalen hålls stabilt låg när knappen inte är nedtryckt.

För Bluetooth kommunikationen används en spänningsdelare med motstånd på 3,3 kΩ och 1,8 kΩ. Syftet är att anpassa signalnivån från Arduino till Bluetooth modulen, eftersom Arduino arbetar med 5 V logik medan Bluetooth modulen använder en lägre signalnivå.

I filter och förstärkarkretsen används motstånd för analog signalbehandling. R1 används tillsammans med C1 för att bilda det första filtersteget vid punkt A. R2 används tillsammans med C2 för det andra filtersteget vid punkt B. R3 och R4 används i återkopplingen till den första LM358 förstärkaren, medan R5 och R6 används på samma sätt i den andra förstärkaren.

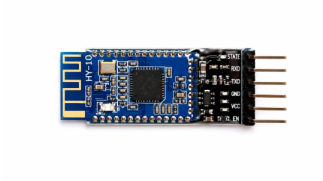
Motstånden R3, R4, R5 och R6 har värdet 10 kΩ. Eftersom förstärkarna är kopplade som icke inverterande förstärkare bestäms förstärkningen av förhållandet mellan två motstånd. När återkopplingsmotståndet och motståndet till GND har samma värde blir förstärkningen ungefär två gånger per förstärkarsteg

4.2.8 SD kortläsare

Figur 4.8: SD kortläsare som används för lagring av mätdata.

SD-kortläsaren används för att spara mätdata efter att mätningen har avslutats, och den kommunicerar med Arduino via SPI. MicroSD-kortet formaterades som FAT32, och resultaten sparas i en textfil, vilket gör det möjligt att öppna filen på en dator och analysera mätvärdena i efterhand. Exempel på lagrade värden är FVC, FEV1 och maxflöde.

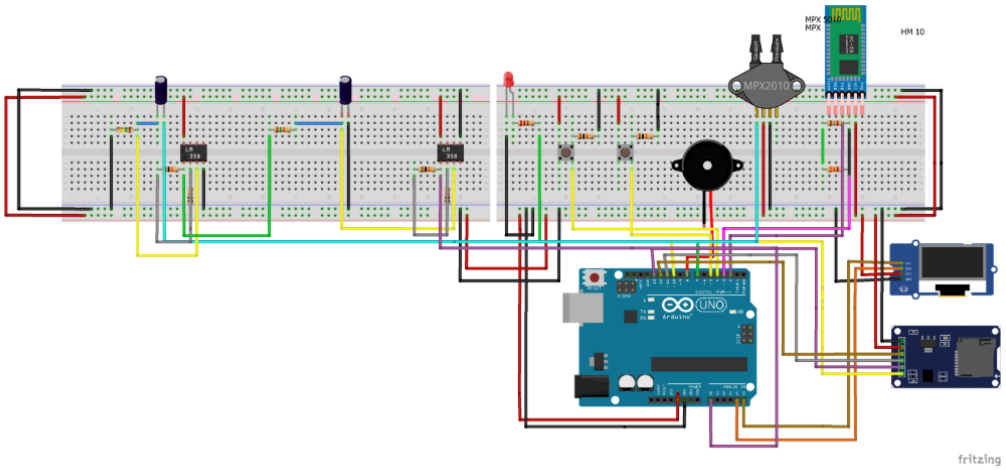
4.2.9 Bluetooth HM 10



Figur 4.9: Bluetooth-modulen HM 10 som används för trådlös dataöverföring.

Bluetooth-modulen HM-10 används för att möjliggöra trådlös överföring av mätdata. Arduino skickar mätvärden seriellt till modulen, varefter HM 10 vidarebefordrar informationen via Bluetooth. För att se till att överföringen fungerade appen BLE Seriel som mottagare under testningen[5].

4.2.10 Komplett systemöversikt



Figur 4.10: Kopplingsschema över den monterade prototypen.

I den monterade prototypen är komponenterna placerade på två kopplingsplattor tillsammans med Arduino Uno. Kopplingsschemat i figur 4.11 visar hur Arduino är ansluten till trycksensorn, OLED-displayen, knapparna, LED, buzzer, SD kortmodulen, Bluetooth modulen samt filter och förstärkarkretsen.

Trycksensorns signal går först via filter och förstärkarkretsen innan den skickas till Arduinos analoga ingång A0. OLED displayen används för att visa mätdata under mätningen, medan knapparna används för att starta och återställa systemet.

SD kortmodulen och Bluetooth modulen är avsedda för lagring respektive trådlös överföring av mätdata. Systemet är konstruerat för att kunna drivas med ett 9 V batteri, men under testning användes främst USB strömförsörjning.

4.2.11 Kopplingsuppbyggnad, breadboard och kablage

Kopplingsplattor, även kallade breadboards, används för att bygga och testa elektroniska kretsar utan lödning. De gör det möjligt att snabbt koppla ihop komponenter, ändra kopplingar och felsöka systemet under utvecklingen. Detta är praktiskt i prototyparbete eftersom komponenter kan flyttas eller bytas ut utan att kretsen behöver byggas om permanent. I prototypen är komponenterna placerade på två kopplingsplattor tillsammans med Arduino Uno. Kopplingsplattorna används för att ansluta trycksensor, OLED display, knappar, LED, buzzer, SD kortmodul, Bluetooth-modul samt filter- och förstärkarkretsen.

Strömmen fördelas via matningsledningarna på kopplingsplattorna. Arduino 5 V kopplas till plusskenan och GND kopplas till minusskenan. Eftersom kretsen är uppbyggd på två kopplingsplattor förs plus och minusskenorna vidare till den andra kopplingsplattan, så att båda delar samma matningsspänning och jord.
Trycksensorn MPX5010 är kopplad till 5 V och GND. Sensorns utsignal går först genom den analoga signalbehandlingskretsen innan den skickas till Arduinos analoga ingång A0. Signalbehandlingen består av kondensatorer, resistorer och två förstärkarsteg med LM358.

Den första LM358 förstärkaren matas med 5 V på pin 8 och GND på pin 4. Signalen från punkt A kopplas till pin 3. Återkopplingen består av två motstånd på 10 kΩ, där R3 kopplas mellan pin 1 och pin 2 och R4 mellan pin 2 och GND. Utsignalen tas från pin 1 och leds vidare till nästa filter- och förstärkarsteg. Den andra LM358

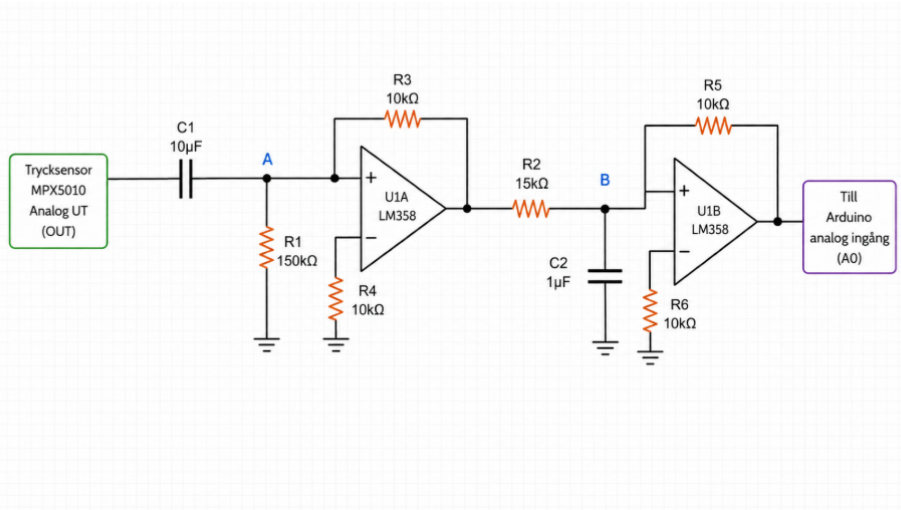
förstärkaren är kopplad på liknande sätt. Den matas också med 5 V och GND. Signalen från punkt B kopplas till förstärkarens ingång. Återkopplingen består av R5 och R6, och den förstärkta utsignalen skickas sedan vidare till Arduinos analoga ingång A0.

På detta sätt behandlas sensorsignalen i två steg innan den används i programvaran. OLED displayen kommunicerar med Arduino via I2C, medan SD kortmodulen kommunicerar via SPI. Knappar, LED och buzzer styrs digitalt från Arduino. Denna uppbyggnad gör att systemet kan testas och justeras stegvis under utvecklingen.

4.2.12 Signalbehandling och filtrering

För att förbättra signalens kvalitet används både analog och digital filtrering. Den analoga signalbehandlingen består av ett bandpassfilter och två förstärkarsteg som byggs med operationsförstärkaren LM358. Den digitala filtreringen sker i programvaran. Målet med dessa åtgärder är att minska brus, dämpa långsamma variationer och förstärka den del av signalen från trycksensorn innan den läses av Arduino. Aktiva RC filter kan byggas med operationsförstärkare, resistorer och kondensatorer för att forma signalens frekvensområde och dämpa oönskade frekvenskomponenter [7].

Signalen från MPX5010 kan innehålla offset, långsamma driftvariationer och högfrekvent brus. Därför behandlas signalen analogt innan den når Arduinos analoga ingång A0. Kretsen är uppbyggd som ett bandpassfilter med två RC steg, följt av två förstärkarsteg med LM358. Kretsens uppbyggnad visas i figuren nedan.



Figur 4.11: Kopplingsschema för filter- och förstärkarkretsen med LM358.

Högpassdelen består av C1 i serie med sensorns utsignal och R1 kopplad från punkt A till GND. Detta steg dämpar offset och mycket långsamma variationer i signalen.

Lågpassdelen sitter mellan förstärkarstegen och består av R2 och C2 vid punkt B. Denna del dämpar högfrekvent brus innan signalen förstärks vidare. Tillsammans bildar högpassdelen och lågpassdelen ett bandpassfilter som släpper igenom de frekvenser som är relevanta vid utandning.

Från punkt A går signalen in i den första LM358 förstärkaren, som är kopplad som en icke-inverterande förstärkare med R3 och R4. Båda motstånden har värdet 10 kΩ, vilket ger en förstärkning på ungefär två gånger. Därefter passerar signalen lågpassdelen vid punkt B och förstärks en gång till i ett andra LM358 steg med R5 och R6. Även dessa motstånd har värdet 10 kΩ, vilket ger en förstärkning på ungefär två gånger. Totalt förstärks signalens variationer ungefär fyra gånger

innan Arduino läser signalen via A0. Den behandlade signalen används sedan i programmet för att uppskatta tryck, luftflöde, FVC och FEV1.

Gränsfrekvenserna räknades fram med RC filterformeln:

$$F_c = \frac{1}{2\pi RC}$$

Där F_c är gränsfrekvensen, R är resistansen i Ω och C är kapacitansen i farad. Högpassdelen består av C1 och R1 och är dimensionerad för att sänka långsamma variationer och offset. Med C1 = 10 μF och en önskad gränsfrekvens på ungefär 0,1 Hz kan R1 räknas ut enligt:

$$R = \frac{1}{2\pi \cdot F_c \cdot C}$$

$$R_1 = \frac{1}{2\pi \cdot 0{,}1 \cdot 10 \cdot 10^{-6}}$$

$$R_1 \approx 159\,k\Omega$$

Eftersom detta värde är inte ett vanligt standardvärde valdes R1 som 150 kΩ. Den faktiska gränsfrekvensen blir:

$$F_{c1} = \frac{1}{2\pi \cdot 150 \cdot 10^3 \cdot 10 \cdot 10^{-6}} \approx 0{,}11\,Hz$$

Lågpassdelen består av R2 och C2 och används för att sänka högfrekvent brus. Och med C2 = 1 μF och en önskad gränsfrekvens på ungefär 10 Hz kan R2 räknas ut enligt:

$$R_2 = \frac{1}{2\pi \cdot 10 \cdot 1 \cdot 10^{-6}} \approx 15{,}9\,k\Omega$$

Eftersom detta värde inte är ett vanligt standardvärde valdes R2 = 15 kΩ. Den faktiska gränsfrekvensen blir då:

$$F_{c2} = \frac{1}{2\pi \cdot 15 \cdot 10^3 \cdot 1 \cdot 10^{-6}} \approx 10{,}6\,Hz$$

Detta betyder att filterdelen dämpar både mycket långsamma variationer och högfrekvent brus, vilket i sin tur gör sensorsignalen mer stabil innan den förstärks och läses av Arduino. Förstärkningen i en icke inverterande operationsförstärkare kan beräknas med följande formel[6]:

$$A_v = 1 + \frac{R_f}{R_g}$$

Där är R_f återkopplingsmotståndet och R_g motståndet mellan den inverterande ingången och GND. Syftet med detta var att varje förstärkarsteg skulle ge en förstärkning på ungefär två gånger. För att få $A_v = 2$ krävs att R_f och R_g har samma värde. A_v den anledningen valdes samma motståndsvärde i varje förstärkarsteg. I första förstärkarsteget används R3 = 10 kΩ. och R4 = kΩ:

$$A_{v1} = 1 + \frac{10\,k\Omega}{10\,k\Omega} = 2$$

Och i andra förstärkarsteget används R5 = 10 kΩ. och R6 = 10 kΩ:

$$A_{v2} = 1 + \frac{10\,k\Omega}{10\,k\Omega} = 2$$

Nu blir den totala förstärkningen:

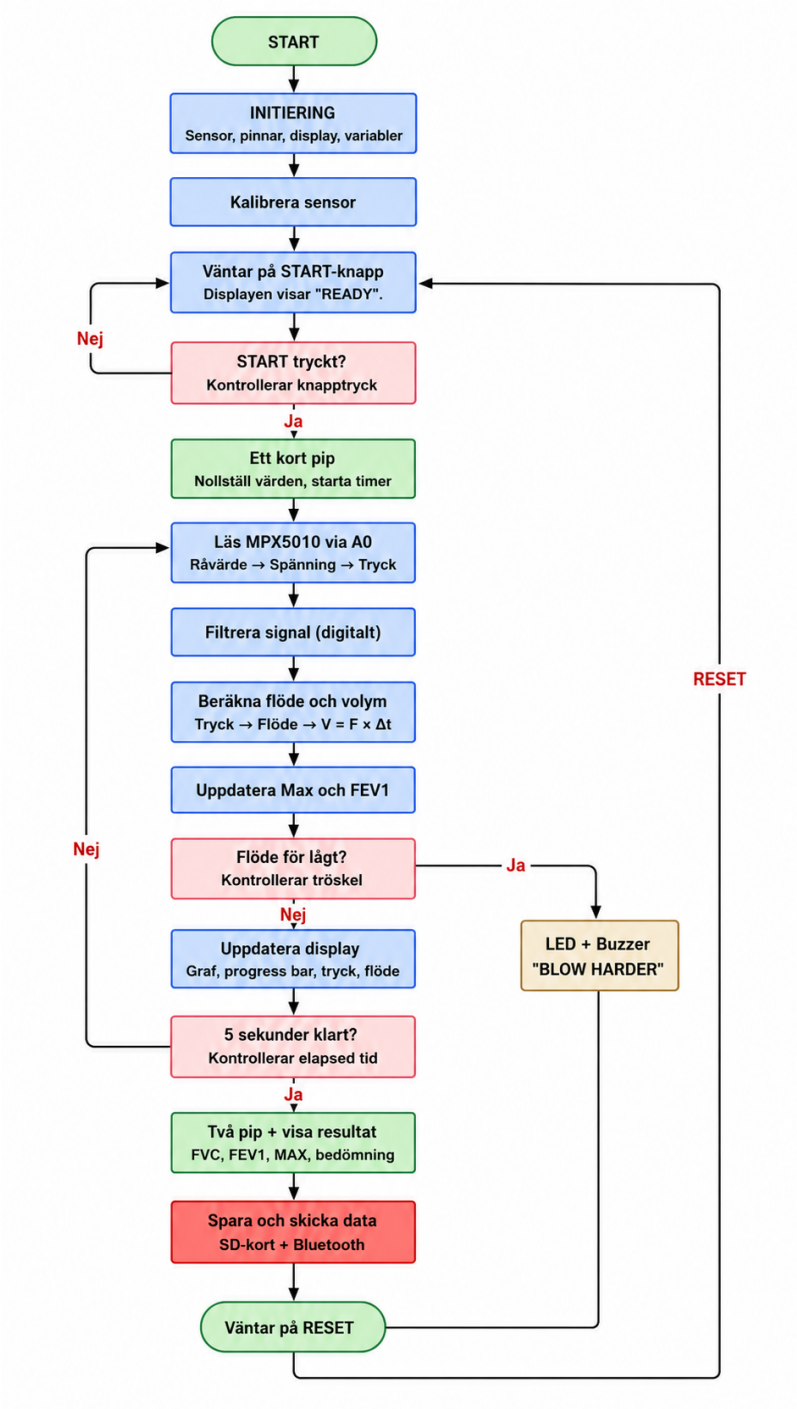
$$A_{v,total} = A_{v1} \cdot A_{v2} = 2 \cdot 2 = 4$$

Detta betyder att signalens variationer förstärks ungefär fyra gånger innan signalen skickas till Arduinos analoga ingång A0. Tabellen nedan visar de komponenter som ingår i filter och förstärkarkretsen samt deras funktion i signalbehandlingen

Tabell 2

Komponent	**Värde**	**Funktion**
C1	10 μF	Del av bandpassfiltrets högpassdel
R1	150 kΩ	Bildar högpassdel tillsammans med C1
R2	15 kΩ	Bildar lågpassdel tillsammans med C2
C2	1 μF	Del av bandpassfiltrets lågpassdel
R3	10 kΩ	Återkopplingsmotstånd i förstärkare 1
R4	10 kΩ	Motstånd till GND i förstärkare 1
R5	10 kΩ	Återkopplingsmotstånd i förstärkare 2
R6	10 kΩ	Motstånd till GND i förstärkare 2
LM358	Operationsförstär	Filtrerar och förstärker sensorsignalen

4.3 Mjukvarudesign



Figur 4.12: Flödesschema över programmets mätsekvens.

Programmet styr systemets funktion från start till avslutad mätning. Vid start initieras sensorn, Arduino pinnarna, OLED displayen och de variabler som används i programmet. Därefter kalibreras sensorn och systemet går till ett READY läge där det väntar på att användaren trycker på startknappen. När mätningen startar ger systemet ett kort pip, nollställer tidigare mätvärden och startar timern. Programmet läser sedan av MPX5010-sensorn via den analoga ingången A0. Råvärdet omvandlas till spänning och tryck, filtreras digitalt och används för att uppskatta luftflöde och volym. Under mätningen uppdateras maxvärden och FEV1. Om

luftflödet är för lågt aktiveras LED och buzzer, samtidigt som displayen visar meddelandet BLOW HARDER". Om flödet är tillräckligt uppdateras displayen med graf, progressbar, tryck och flöde. När mätperioden är klar ger systemet två korta pip och visar resultatvärden som FVC, FEV1 och maximalt luftflöde. Systemet är även avsett att spara data på SD kortet och skicka mätdata via Bluetooth. Efter avslutad mätning väntar systemet på att användaren trycker på resetknappen.

4.4 Användargränssnitt

Användargränssnittet är utformat för att vara enkelt och tydligt vid framtida praktiska tester. Under mätningen visar OLED displayen en realtidsgraf över luftflödet, en progressbar samt värden för aktuellt tryck och flöde. Detta gör att användaren kan följa mätningen medan den pågår. Om utandningen är för svag ger systemet både visuell och hörbar återkoppling. Displayen visar ett meddelande, LED tänds och buzzern ger upprepade pip som varning. Efter avslutad mätning är systemet

avsett att byta till en resultatskärm. Där visas FVC, FEV1 och maximalt uppmätt flöde tillsammans med en enkel kvalitetsbedömning. Bedömningen bygger på om utandningen har varit tillräckligt lång och haft tillräckligt högt flöde.

5 Resultat

I detta kapitel redovisas prototypens funktion vid testtillfällena samt de begränsningar som påverkade mätresultaten. Kapitlet sammanfattar även jämförelsen mellan den utvecklade lösningen och kommersiellt tillgängliga spirometrar.

5.1 Funktionella tester

Prototypens olika delsystem testades steg för steg, bland annat sensorsignal, display, knappar, LED, buzzer, SD-kort och Bluetooth. Alla komponenterna kopplades ihop på breadboard och programlogiken implementerades så att Arduino skulle kunna läsa av sensorsignalen, bearbeta den och visa informationen på displayen. Systemet förbättrades även genom att lägga till återkoppling till användaren i form av meddelandet ”Blow Harder”, en lysdiod och en ljudsignal.

Arbetet genomfördes stegvis, där varje del av systemet prövades separat. Testresultaten visade att de flesta delsystem fungerade individuellt. Under testningen uppstod dock problem med lösa kontakter och instabila signaler, vilket krävde justeringar och felsökning av kopplingarna.

5.2 Noggrannhet och mätdata

Eftersom systemet inte kunde verifieras i sin helhet går det inte att fastställa mätvärdenas tillförlitlighet utan att jämföra mot en klinisk spirometer eller en kontrollerad referensvolym. Dessutom påverkas noggrannheten av flera faktorer, såsom kalibrering, förändringar i Δt, osäkra breadboardkontakter och variationer i luftkanalen. För att uppnå högre tillförlitlighet krävs därför framtida tester mot en etablerad referens.

5.3 Jämförelse med kommersiella lösningar

Till skillnad från prototypen är kommersiella spirometrar avsedda för diagnostisk användning och uppfyller krav på noggrann verifiering. De håller en högre teknisk standard än prototypen som utvecklades i detta projekt.

Tabell 3

Egenskap	**Prototyp**	**Kommersiell spirometer**
Pris	cirka 1300 kr	cirka 4000–25000 kr
Kalibrerad	Nej	Ja
Visar FEV1/FVC	Ja	Ja
Bärbar	Ja	Ja/Nej
Bluetooth	Ja, BLE	Nej/varierar
Datalagring	SD-kort	Internt minne eller molnlagring
Medicinskt godkänd	Nej	Ja

Den utvecklade prototypen kostar ungefär 1300 kr i material och är ett billigt alternativ, men också ett enklare sådant. Det låga priset gör systemet tillgängligt för undervisning och som en demonstration av mätprinciper.

6 Diskussion

Diskussionen utgår från de slutsatser och begränsningar som genererades under projektet, och belyser bland annat tekniska och medicinska aspekter samt faktorer som haft inverkan på systemets stabilitet och noggrannhet.

6.1 Tolkning av resultat

Resultaten visar att det är möjligt att bygga en enkel spirometerprototyp med lättillgängliga komponenter, men steget från fungerande delsystem till stabil mätning är större än vad komponentvalet antyder. Att flera delsystem fungerade var för sig tyder på att systemarkitekturen var rimlig och att mätkedjan kunde byggas upp i både hårdvara och mjukvara.

Trots detta visade arbetet att de största utmaningarna låg i signalkvalitet och systemintegration. Sensorsignalen var inte alltid stabil vilket innebär att både analog och digital filtrering behövdes. En trolig förklaring är att MPX5010 sensorn och slangkopplingarna är känsliga för små mekaniska rörelser, vilket kan skapa variationer i signalen även utan blåsning. Detta påverkade i sin tur möjligheten att verifiera flödes och volymmodellen med stabila mätdata, och därmed är den viktigaste begränsningen i tolkningen av resultatet.

6.2 Begränsningar

Projektets främsta begränsningar var kalibrering, noggrannhet och stabilitet. Eftersom prototypen inte kalibrerades mot en klinisk spirometer eller en känd referensvolym kan värden för tryck, luftflöde, volym, FVC och FEV1 endast betraktas som uppskattningar.

Ett annat problem var att prototypen byggdes med tillfälliga kopplingar på breadboard och med jumperkablar. Detta gav flexibilitet under utvecklingen, men ökade samtidigt risken för glapp och instabila signaler. På liknande sätt var luftkanalen med spruta och slang en förenklad lösning, vilket begränsade möjligheten att skapa helt repeterbara mätförhållanden.

Utöver detta påverkades noggrannheten också av tidssteget Δt, eftersom volymberäkningen bygger på att flödet summeras över tid. Om tiden mellan avläsningarna varierar påverkas därmed även beräkningen av volym, FVC och FEV1. En möjlig orsak till detta är att programmet hanterar sensoravläsning, displayuppdatering, och kommunikation i samma loop, vilket gör att looptiden kan variera.

6.3 Etiska och medicinska aspekter

Prototypen är ett demonstrations- och utbildningssystem och får därför inte användas för medicinsk diagnostik. Eftersom systemet varken är kalibrerat eller validerat finns en risk att felaktiga värden tolkas som tillförlitliga om prototypen presenteras i fel sammanhang. Det är därför viktigt att tydligt kommunicera att de uppskattade värdena för FVC och FEV1 inte är diagnostiska och inte kan ersätta en klinisk spirometriundersökning.

6.4 Framtida arbete

Nästa steg bör vara att vidare testa systemet med slang- och sprutuppställningen under mer stabila förhållanden. Samtidigt bör systemet kalibreras mot en referensspirometer, så att uppskattningarna för flöde, FVC och FEV1 kan justeras och bedömas mer tillförlitligt.

På mjukvarusidan bör tidssteget Δt mätas med millis() eller micros() istället för att följa looptiden. Detta skulle framför allt förbättra FEV1 beräkningen. Vidare bör breadboard och jumperkablar ersättas med lödda kopplingar eller ett PCB för att få stabilare signaler och minska risken för de kontaktproblem som uppstod under projektet.

På längre sikt vore en egen mobilapp för Bluetooth överföringen ett naturligt tillägg. I samband med detta behöver även datasäkerheten ses över, eftersom mätdata då kan kopplas till enskilda personer [5].

Vid eventuell vidareutveckling för hantering av mätdata om enskilda personer måste även dataskydd beaktas. Eftersom mätdata överförs trådlöst via Bluetooth kan överföring av hälsorelaterad information medföra säkerhetsrisker som behöver hanteras vid en eventuell fortsatt utveckling[5]. För den nuvarande prototypen är dessa risker däremot begränsade, eftersom den används för tekniska demonstrationer utan koppling till verkliga patientdata. Det bör dock beaktas tidigt om systemet ska användas med riktiga personer.

7 Slutsats

Projektet resulterade i en Arduino baserad spirometerprototyp som demonstrerar mätkedjan från trycksignal till uppskattade lungfunktionsvärden. Delsystemen verifierades, men prototypen behöver fortsatt stabilisering och kalibrering innan tillförlitliga mätvärden kan tas fram. En av de viktigaste lärdomarna från projektet är att breadboard och jumperkablar inte är tillräckligt pålitliga när många komponenter ska kopplas i ett mätsystem. Eftersom kontaktproblemen var återkommande och svåra att felsöka, begränsades vad som faktiskt kunde verifieras. Detta problem bör hanteras tidigt vid vidareutveckling, exempelvis genom att använda lödda kopplingar eller ett kretskort (PCB).

Dessutom, om prototypen ska användas i undervisning, är det troligaste hindret inte komponenterna i sig utan snarare kalibreringen och stabiliteten. Utan att jämföra med en känd volym är det svårt att avgöra om mätvärdena är korrekta. Därmed blir prototypen mindre lämplig för undervisning i lungfunktion. Därför kan okalibrerad prototyp visa värden som ser rimliga ut, men som inte säkert stämmer överens med faktiska värden för lungfunktion. Detta kan i sin tur leda till missförstånd hos användaren.

Referenser

1. P. Sridevi, P. Kundu, T. Islam, C. Shahnaz och S. A. Fattah, "A Low-cost Venturi Tube Spirometer for the Diagnosis of COPD," i *TENCON 2018 – 2018 IEEE Region 10 Conference*, Jeju, Sydkorea, 2018. Hämtad: 2026-05-14.

   Länk: https://doi.org/10.1109/TENCON.2018.8650092

2. P. Zhou, L. Yang och Y.-X. Huang, "A Smart Phone Based Handheld Wireless Spirometer with Functions and Precision Comparable to Laboratory Spirometers," *Sensors*, vol. 19, nr. 11, artikel 2487, maj 2019. Hämtad: 2026-05-14.

   Länk: https://pmc.ncbi.nlm.nih.gov/articles/PMC6603793/

3. S. Ilman och E. Wahyuningsih, "Portable Spirometer Using Air Pressure Sensor MPX5500DP Based on Microcontroller Arduino Uno," i *Proceedings of The International Conference on Environmental and Technology of Law, Business and Education on Post Covid 19*, ICETLAWBE 2020, Bandar Lampung, Indonesien, 2020. Hämtad: 2026-05-14.

   Länk: http://repository.lppm.unila.ac.id/50343/1/ICETLAWBE.pdf

4. M. F. Nunes, H. P. da Silva, L. Raposo och F. Rodrigues, "Design and Evaluation of a Novel Venturi-Based Spirometer for Home Respiratory Monitoring," *Sensors*, vol. 24, nr. 17, artikel 5622, aug. 2024. Hämtad: 2026-05-14.

   Länk: https://www.mdpi.com/1424-8220/24/17/5622

5. G. A. Zendehdel, R. Kaur, I. Chopra, N. Stakhanova och E. Scheme, "Automated Security Assessment Framework for Wearable BLE-enabled Health Monitoring Devices," *ACM Transactions on Internet Technology*, vol. 22, nr. 1, artikel 14, sep. 2021. Hämtad: 2026-05-14.

   Länk: https://doi.org/10.1145/3448649

6. C. Wells och J. Becker, "Low-Cost Digital Programmable Gain Amplifier Reference Design," Texas Instruments, TIDUB13, nov. 2015. Hämtad: 2026-05-14.

   Länk: https://www.ti.com/lit/ug/tidub13/tidub13.pdf

7. E. J. Foster, "Active Low-Pass Filter Design," *IEEE Transactions on Audio*, vol. AU-13, nr. 5, s. 104–111, sep./okt. 1965. Hämtad: 2026-05-14.

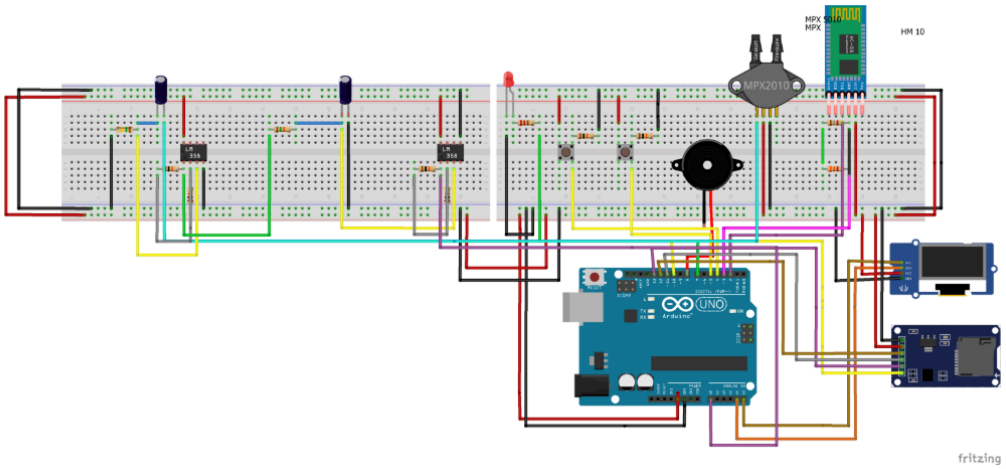
   Länk: https://ieeexplore.ieee.org/document/1161808

Bilagor

Bilaga A – Arduino-kod

Projektets kod finns tillgänglig via GitHub repository:
`https://github.com/karam-king/Utveckling-av-en-smart-spirometer/blob/main/smart_spirometer.ino`
Denna kod används för att läsa sensorsignal från MPX5010, bearbeta signalen, uppskatta tryck, luftflöde, volym, FVC och FEV1 samt visa resultaten på OLED displayen.

Bilaga B – Kopplingsschema



Bilaga C – Komponentlista och pinlista

Tabell 4. Komponentlista

Komponent	Värde/Typ	Funktion
Arduino Uno	Mikrokontroller	Läser sensorsignal och styr systemet
MPX5010	Trycksensor	Mäter tryckförändringar
OLED-display	I2C-display	Visar mätvärden, graf och status
SD-kortmodul	SPI-modul	Sparar mätdata
Bluetooth-modul	HM-10	Skickar information trådlöst
LM358	Operationsförstärka	Förstärker sensorsignalen
LED	Lysdiod	Ger visuell återkoppling
Buzzer	Summer	Ger hörbar återkoppling
Startknapp	Tryckknapp	Startar mätningen
Resetknapp	Tryckknapp	Återställer systemet
C1	10 μF	Del av högpassdelen i filtret
C2	1 μF	Del av lågpassdelen i filtret
R1	150 kΩ	Bildar högpassdel tillsammans med C1
R2	15 kΩ	Bildar lågpassdel tillsammans med C2
R3	10 kΩ	Återkopplingsmotstånd i förstärkare 1
R4	10 kΩ	Motstånd till GND i förstärkare 1
R5	10 kΩ	Återkopplingsmotstånd i förstärkare 2
R6	10 kΩ	Motstånd till GND i förstärkare 2
LED-motstånd	220 Ω	Begränsar strömmen till LED
Pull-down-motstånd	10 kΩ	Stabiliserar knappsignaler
Spänningsdelare	3,3 kΩ och 1,8 kΩ	Anpassar signalnivå till Bluetooth

Tabell 5. Pinlista

Modul/Komponer	**Pin på kompo-nent**	**Kopplas till**
MPX5010	VOUT	Filterkrets och Arduino A0
MPX5010	VCC	5 V
MPX5010	GND	GND
OLED-display	VCC	5 V
OLED-display	GND	GND
OLED-display	SDA	Arduino A4
OLED-display	SCL	Arduino A5
SD-kortmodul	VCC	5 V
SD-kortmodul	GND	GND
SD-kortmodul	CS	Arduino pin 10
SD-kortmodul	MOSI	Arduino pin 11
SD-kortmodul	MISO	Arduino pin 12
SD-kortmodul	SCK	Arduino pin 13
Bluetooth HM-10	VCC	5 V
Bluetooth HM-10	GND	GND
Bluetooth HM-10	TXD	Arduino pin 2
Bluetooth HM-10	RXD	Arduino pin 3 via spänningsdelare
LED	Anod	Arduino pin 7 via 220 Ω
LED	Katod	GND
Buzzer	+	Arduino pin 8
Buzzer	-	GND
Startknapp	Signal	Arduino pin 4
Resetknapp	Signal	Arduino pin 5

Tabell 6. Filter och förstärkare krets

Del	**Koppling**
C1 långa ben, +	MPX5010 Signal/VOUT
C1 korta ben, –	Punkt A
R1	Punkt A till GND
Punkt A	Förstärkare 1, pin 3
LM358, pin 8	5 V
LM358, pin 4	GND
R3	Mellan pin 1 och pin 2 på förstärkare 1
R4	Mellan pin 2 och GND på förstärkare 1
LM358 förstärkare 1, pin 1	Till R2
R2	Mellan förstärkare 1 pin 1 och punkt B
C2 långa ben, +	Punkt B
C2 korta ben, –	GND
Punkt B	LM358 förstärkare 2, pin 3
R5	Mellan pin 1 och pin 2 på förstärkare 2
R6	Mellan pin 2 och GND på förstärkare 2
LM358 förstärkare 2, pin 1	Arduino A0

INSTITUTIONEN FÖR DATA- OCH INFORMATIONSTEKNIK
MEKATRONIK OCH ELEKTROTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2026 www.chalmers.se

CHALMERS