# Progress Report CW8

## Start of the bachelor thesis

The advancement of micro- and nanoscale force sensing technologies has opened new possibilities for precision measurements in biophysics, material science, and nanotechnology. One such innovation is the self-excited FluidFM cantilever system, which combines microfluidic control with atomic force microscopy (AFM) principles. The Laboratory of Biosensors and Bioelectronics (LBB) at ETH Zurich has been developing this system to enhance stability, sensitivity, and automation in force measurements.

This thesis focuses on improving the self-excited FluidFM cantilever system by optimizing its excitation and control mechanism. The objective is to refine its performance, accuracy, and potential applications in dynamic force sensing. By addressing key challenges such as signal stability, resonance behavior, and feedback control, this work aims to contribute to the broader goal of enhancing real-time force measurements with the FluidFM.

## Introduction to Fluidic Force Microscopy (FluidFM)

Fluidic Force Microscopy (FluidFM) is an advanced technology that combines the precision force measurement capabilities of atomic force microscopy (AFM) with microfluidics. In FluidFM, a hollow cantilever with an integrated microchannel enables the direct handling and manipulation of fluids at the microscale. This integration allows for applications such as single-cell injection, nano-printing of biomolecules, and localized substance delivery or extraction in a controlled manner.

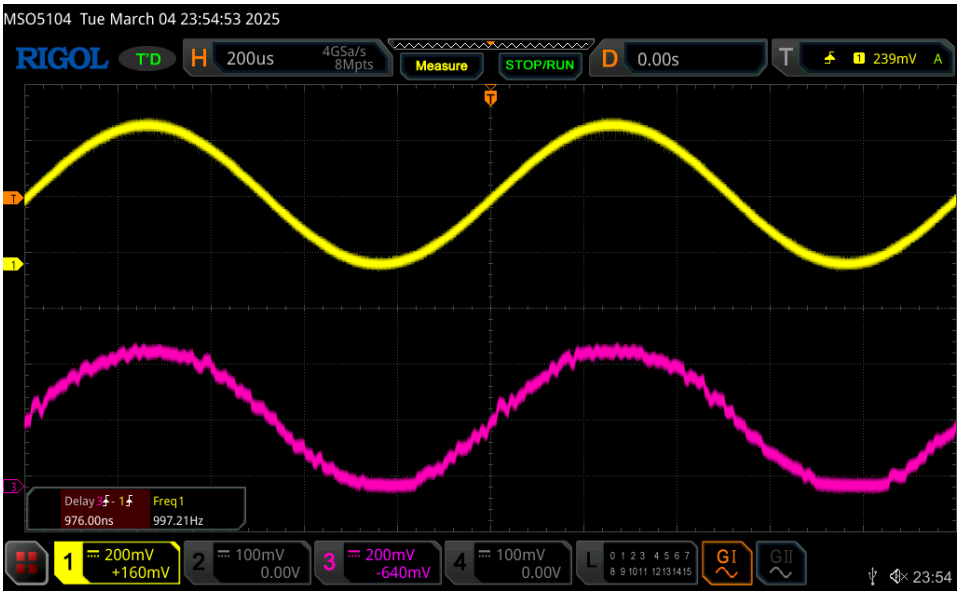


Figure 1: Schematic Representation of the FluidFM Setup.

## Working principles of FluidFM

Figure 1 illustrates the working principle of the FluidFM system, which integrates microfluidics with an atomic force microscope (AFM) for precise force measurements and liquid manipulation. The system consists of a FluidFM microfluidics control unit, which supplies fluid through tubing to a cantilever with an internal microchannel. The carrier clip holds the cantilever, which is connected to a fluid reservoir that enables controlled dispensing of liquids.

The AFM scan head contains a cantilever holder and a laser detection system. The laser beam reflects off the cantilever and is used to measure its deflection, providing force-sensitive feedback. The cantilever is positioned above a substrate, which can be a biological sample or another surface of interest. The inset zooms into the cantilever tip, highlighting the internal fluid channel that allows precise liquid handling.

This setup enables applications such as single-cell manipulation, nanolithography, and force spectroscopy, making FluidFM a powerful tool in nanotechnology and biophysics.

## Advantages of FluidFM

- **Precise Force Control** FluidFM retains the force measurement functionality of AFM, enabling high-precision force spectroscopy. This is crucial for studying cell mechanics, adhesion properties, and other biomechanical interactions at the single-cell or even subcellular level.
- **Versatile Fluid Handling** The microchannel inside the cantilever allows for the controlled injection or aspiration of minute fluid volumes. Researchers can deliver biomolecules, drugs, or other solutions directly into cells or tissues, making it invaluable for targeted experimental studies.
- **Reduced Sample Damage** Compared to conventional force microscopy or other micromanipulation techniques, FluidFM can apply lower forces and smaller volumes of liquid. This decreases the risk of damaging sensitive samples, such as living cells or delicate structures.
- **Broad Application Range** The technology is used in various fields, including cell biology, materials science, and biosensor development. Its unique combination of force measurement and fluid handling paves the way for innovative research methods, such as intracellular measurements, cell sorting, and 3D bioprinting.

In summary, FluidFM offers a powerful and versatile platform for both researchers and industrial applications. By integrating microfluidic channels with force-sensing cantilevers, scientists can precisely manipulate minute volumes of liquid and measure microscale forces, providing valuable insights into biological, chemical, and materials processes.

## Introduction to the Problem

Achieving first-order resonance in the Fluid-FM cantilever is crucial because it generally provides the highest amplitude response, enhancing measurement sensitivity and control in fluidic environments. However, the resonance frequency depends strongly on the droplet's mass at the tip, which varies by size and density from one experiment to another. As a result, the cantilever may inadvertently excite higher-order modes instead of the intended first-order resonance. This variability necessitates recalibrating the resonance frequency for each droplet configuration to ensure optimal performance.

## Research Motivation

FluidFM is a powerful tool for nanotechnology, biophysics, and single-cell biology. Its ability to manipulate cells non-invasively, perform highly controlled liquid dispensing, and conduct force spectroscopy makes it superior to traditional AFM in many biomedical applications. Addressing this issue would significantly enhance FluidFM's reliability and effectiveness, amplifying its impact across biomedical applications.

## Current Approach

problems:

1. higher resonance frequencies, using a low pass filter
2. unstable system, (goes to infinity or maximum) amplitude threshold filter
3. excitement, the threshold filter introduces a new problem it prevents excitement of the oscillation in the beginning. Just applying a step function does not solve it


Figure 2: Informal schematic oversimplified representation of a Fluid-FM and the device requirements which are to be designed in the bachelor thesis.

## Towards a Solution: Developing a logical Filter

digital (logical) filters provide higher accuracy, flexibility, and stability, making them the preferred choice in most modern applications, especially where precision and adaptability are required.


Figure 3: Informal schematic oversimplified representation of a Fluid-FM and the device requirements which are to be designed in the bachelor thesis.

## Technical requirements

high sampling rate, high resolution DAC, ADC, two cores, networking, coxial input power = usb c => data to computer ethernet maybe sd-card?

## Hardware Selection

1. Arduino Pro™ Portenta H7 REV2
2. Arduino Pro™ Portenta Max Carrier
3. Arduino Pro™ Portenta Breakout

## Issues

Issues I ran into using the Portenta H7 Ordering the mid carrier

# Proposed Section for Bachelor Thesis

One of the initial ideas for prototyping involved using the Arduino Portenta H7 board. Although this board can be programmed with the Arduino IDE, it poses several challenges for professional embedded development. First, the Arduino ecosystem lacks a robust debugging interface, making it difficult to perform efficient troubleshooting during the development phase. Moreover, Arduino uses its own variant of C/C++, which can be tedious to work with when implementing more advanced features.

An attempt to program and flash the Portenta H7 using Rust highlighted further limitations. While Rust is rapidly growing in the embedded domain and offers advantages in terms of memory safety, performance, and tooling, uploading code to the Portenta proved cumbersome. Tools such as `dfu-util` and tutorials (e.g., those provided by Amazon Zephyr) offered partial guidance, but stable and straightforward Rust support remained elusive on the Portenta platform.

## Limitations of Off-the-Shelf Carrier Boards

The Portenta family also offers carrier boards with various connectors (including coaxial connectors). However, these are often tied to specific modules—such as LoRa or GSM modules—not necessarily aligned with the custom requirements of this project. Additionally, the Portenta Max Carrier board, as well as the Portenta H7 itself, come with numerous components that are superfluous for a highly specialized application. This increases the overall cost and complicates the hardware setup, as most of these features remain unused.

## Rationale for a Custom Board

Given the challenges encountered with the Portenta ecosystem and the need for a tailored hardware solution, designing a custom printed circuit board (PCB) emerges as the most viable strategy. The key advantages of a custom PCB include:

- **Control Over Hardware Design** By selecting only the necessary components for signal filtering, coaxial input, and data processing, the board can be optimized for size, cost, and noise performance. Unused features—common in many development boards—are avoided.
- **Improved Debugging and Development Environment** A custom board allows for seamless integration of an external debugger and supports a broader range of software tools. Developers can choose environments such as Rust or standard C/C++ with full debugging capabilities, a critical requirement for reliable and efficient embedded systems development.
- **Microcontroller Selection** The STM32H747XIHx microcontroller offers promising features, particularly for this application. Its dual-core architecture allows one core to handle peripheral communication (e.g., controlling fluidic systems, networking, LabVIEW interfacing) while the other core processes and filters the incoming signal data. Additionally, this family of MCUs boasts high-resolution, high-speed ADCs that fit the project's need for accurate and fast signal sampling. But is there another microcontroller from perhaps the same family that offers even better specs? Yes, indeed there is.
- **Cost-Effectiveness at Scale** For professional or commercial applications, the per-unit cost of custom boards often becomes more attractive compared to using feature-heavy development boards. By sourcing components through manufacturers like JLCPCB, assembly can be consolidated into a single service, simplifying logistics and reducing potential errors in manual assembly.
- Change the chip, why pick kicad 9?

first schemetic proposal

## Implementation Plan

1. **Initial Prototyping** Short-term testing and validation of essential concepts can still be performed using the Portenta H7 on a simple breadboard. This step confirms software viability and basic signal acquisition before committing to a final PCB design.
2. **PCB Design** Once the feasibility has been validated, the custom board layout will be designed to incorporate the STM32H747XIHx MCU, necessary power regulation, and coaxial connectors for clean signal inputs. Specialized debugging interfaces (e.g., SWD or JTAG) will also be included.

3. **Manufacturing and Assembly** The board design files will be sent to a manufacturer (e.g., JLCPCB), which provides both PCB fabrication and component assembly services. This ensures a streamlined supply chain and consistent quality.
4. **Software Development and Integration** Leveraging the Rust ecosystem—or standard C/C++ with a professional IDE—will offer a more robust development workflow, with advanced features such as breakpoints, real-time debugging, and better resource management.

In conclusion, while the Arduino Portenta H7 can serve as a convenient development platform for preliminary tests, its limitations in debugging, the complexity of customizing it for Rust, and extraneous hardware components make it less suitable for this project's professional and specialized requirements. A custom STM32H747XIHx-based PCB addresses these drawbacks by providing a tailored, cost-effective, and high-performance solution.

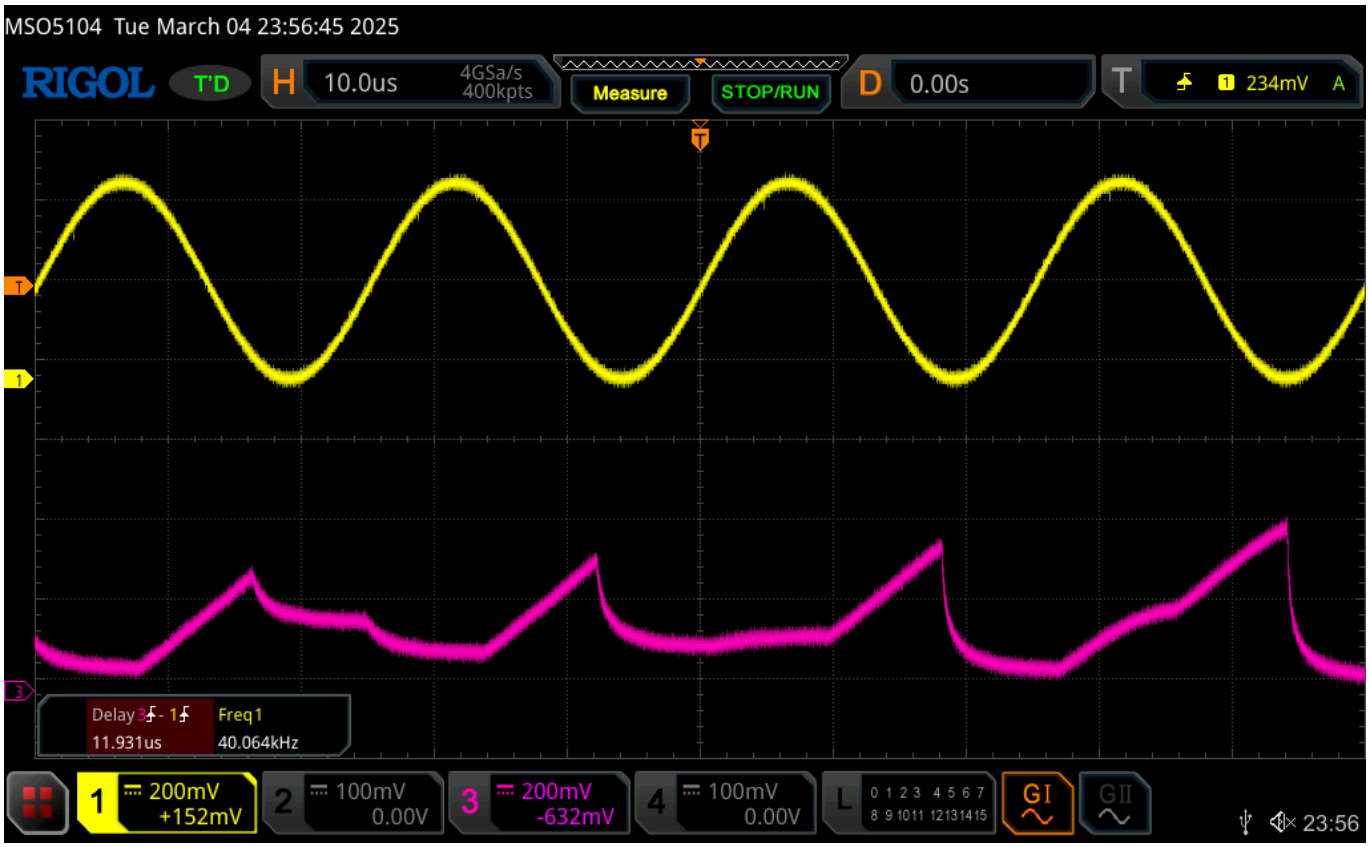


Figure 4: First schematics of the synthesizer

```c
#include "Arduino.h"
#include "hal/analogout_api.h"
#include "hal/analogin_api.h"

// Our objects
dac_t myDac;
analogin_t myAdc;

void setup() {

  // Convert the Arduino pin name (e.g. A0) to an Mbed pin name
  PinName dacPin = digitalPinToPinName(A6);  // Suppose your board says A0 is a DAC
  PinName adcPin = digitalPinToPinName(A1);  // Suppose A1 is a regular analog input

  // Initialize the DAC on A0
  analogout_init(&myDac, dacPin);

  // Initialize the ADC on A1
  analogin_init(&myAdc, adcPin);
}
```

```
21
22 void loop() {
23   // Read a 16-bit value from the ADC
24   //uint16_t adcValue = analogin_read_u16(&myAdc);
25   float adcValue = analogin_read(&myAdc);
26   // Write that value to the DAC
27   analogout_write(&myDac, adcValue);
28 }
```

## Bibliography