OULUN YLIOPISTO
TIETO- JA SÄHKÖTEKNIIKAN TIEDEKUNTA
ELEKTRONIIKAN JA TIETOLIIKENNETEKNIIKAN TUTKINTO-OHJELMA

BACHELOR'S THESIS

Controlling cooling fans with Arduino Pulse Width Modulation controller

Author Aapeli Huttunen

Supervisor Jari Hannu

February 2023

Huttunen A. (2023) Controlling cooling fans with Arduino Pulse Width Modulation controller. University of Oulu, Degree Programme in Electronics and Communications Engineering. Master's Thesis, 20 p.

ABSTRACT

In this thesis a fan controller was implemented using an Arduino to supply a pulse width modulation (PWM) control signal. The controller is used to control fans when there is no control signal and power available from a finished base transceiver station. This Bachelors thesis consists of the planning, sourcing, building, developing software and testing the implemented controller.

The controller was built with an Arduino board and 48V power supply which are controlled via a PC. Two different types of software were used, a simpler one using Windows Remote Arduino Experience and a Python based software which enabled more precise control.

Arduino, Pulse Width Modulation, PWM, Cooling fans.

Huttunen A. (2023) Jäähdytyspuhaltimien ohjaus Arduino pulssinleveysmodulaatio signaalilla. Oulun yliopisto, Tieto- ja sähkötekniikan tiedekunta, Elektroniikan ja tietoliikennetekniikan tutkinto-ohjelma. Kandidaatintyö, 20 s.

TIIVISTELMÄ

Tässä kandidaattityössä toteutettiin Arduino-pohjainen pulssileveysmodulaatiolla ohjattava puhallinpaketti. Ohjainta käytetään ohjaamaan kehitysvaiheessa olevien tukiasemalaitteiden puhaltimia. Kandidaattityö koostuu ohjaimen suunnittelusta, hankinnasta, kasaamisesta, ohjelmoimista ja testauksesta.

Ohjain koostuu Arduinosta ja 48V virtalähteestä. Arduinoa, joka generoi kanttiaaltoa, ohjataan tietokoneella. Käytössä on kaksi eri ohjelmistoa, yksinkertaisempi Windows Remote Arduino Experience sekä Python-pohjainen kehittäjäohjelmisto, joka mahdollistaa tarkemman ohjauksen.

Avainsanat: Arduino, pulssinleveysmodulaatio, puhallin.

TABLE OF CONTENTS

FOREWORD

The purpose of this bachelor's thesis was to create fan control devices for Nokia.

I want to thank Jari Hannu for supervising this bachelor's thesis. I also want to thank Nokia for providing the topic and materials to build the project. Thank Jarno Rautio and Juha Vuorma for helping build the devices. Special thanks to Essi, Jari and Sari for motivating me and helping throughout the project.

Oulu 15.2.2023

Aapeli Huttunen

LIST OF ABBREVIATIONS AND SYMBOLS

3G	Third generation of wireless mobile telecommunication technology
4G	Fourth generation of wireless mobile telecommunication technology
5G	Fifth generation of wireless mobile telecommunication technology
RAT	Radio access technology
BB	Base band
BTS	Base transceiver station
DC	Direct current
PC	Personal computer
PWM	Pulse width modulation.
RPM	Revolutions per minute
USB	Universal serial bus
V	Volts
P	Power
U	Voltage
I	Current
R&D	Research and development

1 INTRODUCTION

A base transceiver station (BTS) is a part of the mobile network which communicates with the end user and connects the user to the internet. Telecommunication system is illustrated in Figure 1 and base transceiver station is shown in the circle.

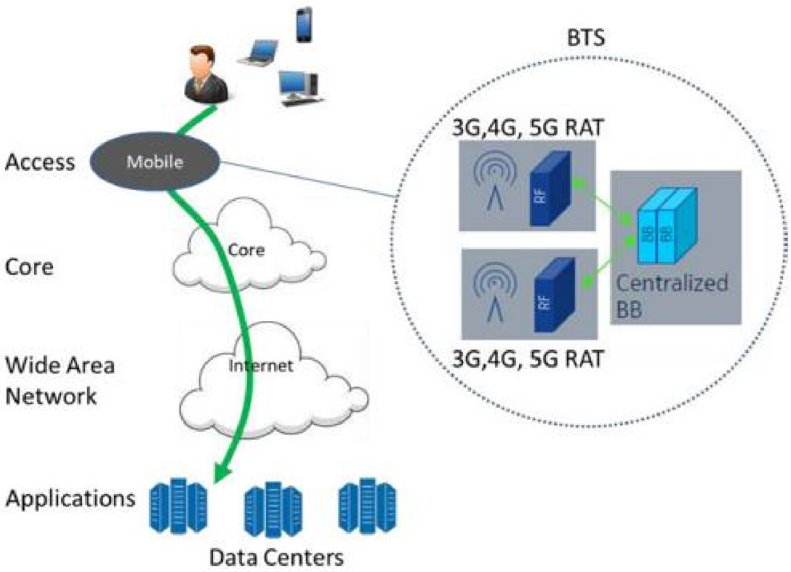

Figure 1. Illustration of a telecommunication system. [1]

A BTS requires cooling, which can be achieved by using fans to exhaust hot air out while bringing cool air through the heatsink. Testing fans is important for both cooling and to keep acoustics noise in control. The base station must be cooled adequately, but without the fans producing too much acoustics noise. Commonly fans use 12 V, but 24 V and 48 V are also used as the operating voltage.

This thesis consists of the planning, sourcing, building, developing software and testing the implemented controller. In this project fans with 48 V power supply were used which are controlled by Pulse Width Modulation (PWM) signal. The Arduino based PWM controller is used to control fans in base station products which are in development and do not necessarily have the needed software or hardware for the fan control. Hence there was a need for a simpler, more transportable, and lower cost control device which could be used instead of more expensive and harder to transport signal generators. The cost can be reduced significantly. With a readymade circuit the Arduino is easy to use even for new users. Additionally, a python script was created to allow more precise control and information for development use.

2 HARDWARE DESIGN AND IMPLEMENTATION

The project was to create Arduino based fan controllers which can output a PWM control signal for a fan assembly that uses 5 V control voltage. Two types of controllers were made, one for demonstrating fan noise and one for laboratory use which has more accurate control. The demo unit is meant to be used as a standalone unit and the fan assembly is attached to a stand shown in Figure 2. This was built to control different fan speeds and demonstrate the approximate acoustics noise level a fan cooled base station can make and its cooling capabilities. The other unit was made for laboratory use with products that are in development and do not have ready software or hardware to control the fans. In this unit the fan assembly is attached to the base station product. This is needed to speed up the testing phase and can be used to control fan speed in noise measurements where powering on the radio product could be challenging or not possible.

An Arduino Uno is a single board microcontroller with the ability to output and input analog and digital signals. [2] Arduino is controlled and powered with a computer. The circuit layout can be seen in Figure 3. The fans are powered with constant 48 V direct current (DC), and the speed is adjusted via PWM signal. The Arduino is controlled with a Personal computer (PC) with either windows remote Arduino experience or python script with more control options.

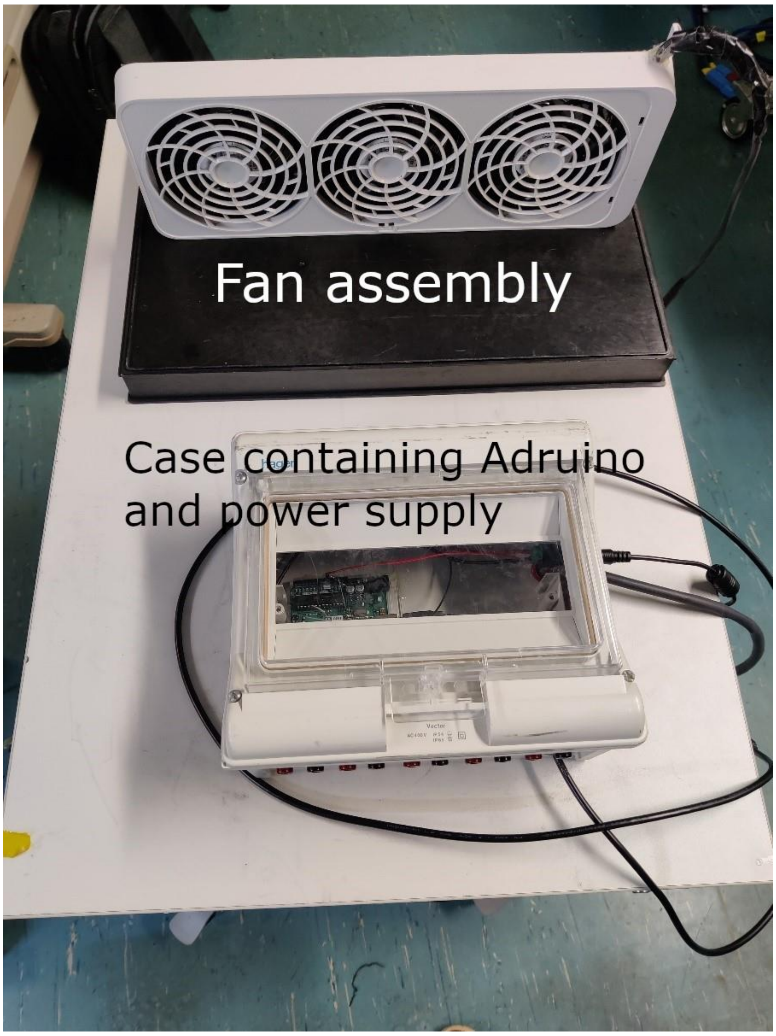

Figure 2. Fan assembly and case containing Arduino and 60W power supply. (Photo Aapeli Huttunen)

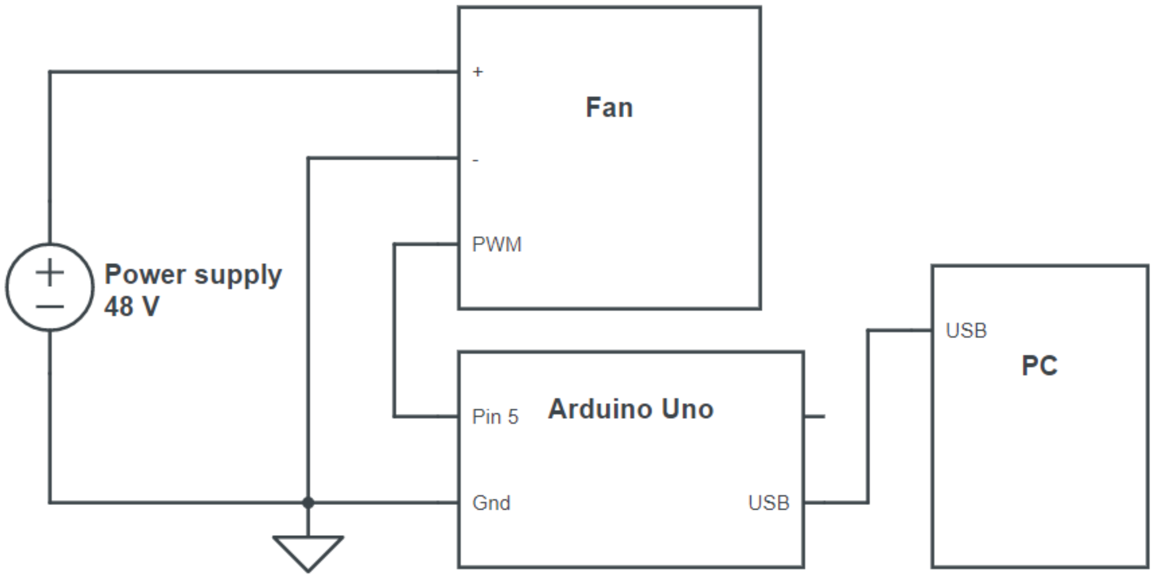

Figure 3. PMW controller connections. Universal Serial Bus (USB) is removable from PC, but other connections are permanent.

2.1 The use cases for the controller

Two units were made for two different use cases. One was made to be used in BTS product and their noise level demonstrations and this has a built-in power supply and easy to use software. The other unit was made for research and development (R&D) testing and this one had preset values and an option to input any duty cycle within limits. Arduino was chosen as the platform because of its ability to generate 5 V PWM signal. Additionally, the Arduino devices are easily available and relatively cheap.

The demo fans assembly is a compact and easy to use package. The main requirement for the software used to control the Arduino is that it must be simple and reliable. This unit only requires minimal setup as in plugging it in to power and USB. For the software Arduino Remote Experience for Windows was chosen since it is easy for anyone to download and install. The board itself does not require setup since it has Arduino standard Firmata on it and the PC side only outputs control values to the board. All the wiring is wrapped in this unit and additional finger guards are installed on the back side of the assembly for safety.

For R&D testing the fan controller should have several ways to control the fan assembly. Presets can be used for quick and easy use, but exact control also must be available. This unit does not have a built-in power supply but has power cables which can be connected to a bench power supply. The fan packet can be swapped out if other fans are required, and the assembly can be attached to where it is necessary.

2.2 Pulse width modulation - PWM

Pulse width modulation (PWM) is a square wave which switches between two values at a fast rate. In this project the PWM is switching between 0 V and 5 V. The signal is on for a certain percentage of the signal period and that is the duty cycle. The duty cycle changes the fans speed by switching the fan power on and off. PWM is well suited for controlling fans because it has more accurate control and can be used to get less revolutions per minute (RPM). RPM is the number of full cycles the fan spins in a minute.

In Figure 4 a) the PWM signal is on for 90% of the time so this means the fans 48 V power is switched on for 90% of the time. The red lines represent the percentage of time the power is on in each scenario. In Figure 4 b) the duty cycle is 50% and that has the 48 V power on for 50% of the time. In Figure 4 c) the duty cycle is 10%. 10% duty cycle is not enough to make the fan spin, it must be around 30% duty cycle to start spinning at 800 RPM which is the minimum RPM the fan can spin at.

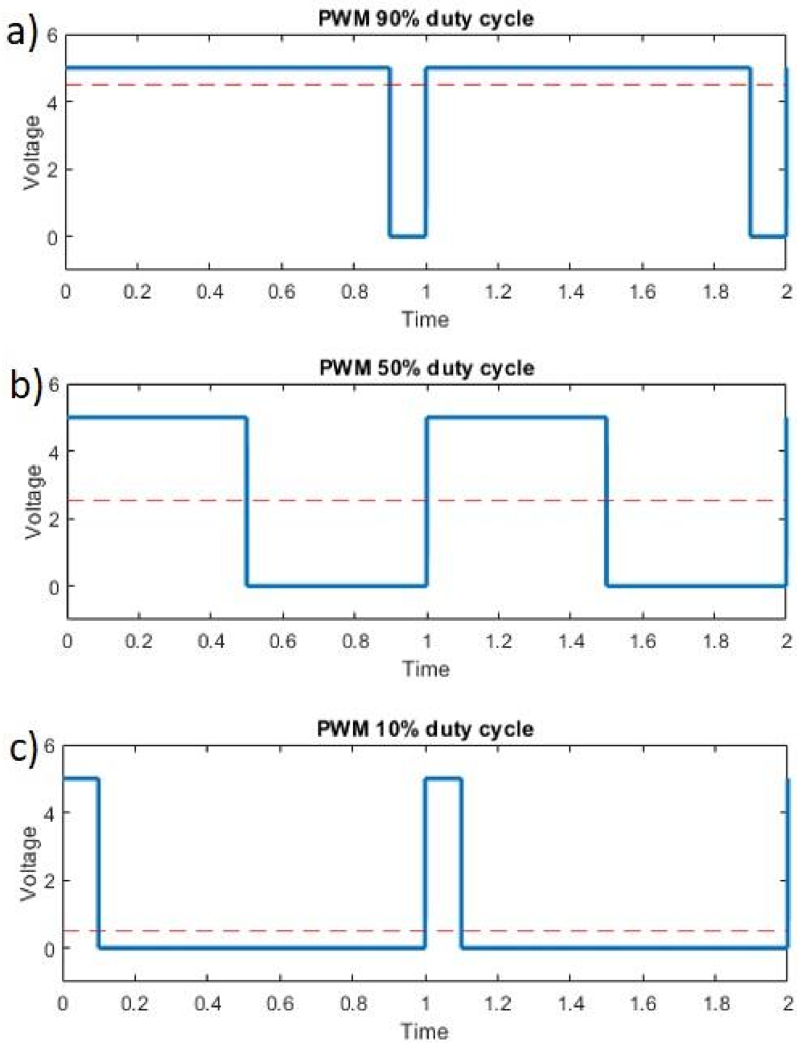

Figure 4. a) The PWM signal duty cycle is 90%. **b)** The PWM signal duty cycle is 50%. **c)** The PWM signal duty cycle is 10%. The red line represents the time the signal is at 5 V.

The fans used are reverse controlled and are at full speed when the PWM duty cycle is at 0 percent and stop at hundred percent. The fans used work with 1 kHz PWM signal, and an Arduino can output this from certain pins.

2.3 System design

The components used are an Arduino Uno revision 3, jumper cables, power supply, case, and fan assembly. Also, a USB and power cable were needed. For the unit used in demos the Arduino and power supply had to be in one portable case with the fan assembly on a separate stand. To make the fans safe fingerguards were used on the side with no existing protection.

Arduino uno was used since it can easily output 1kHz PWM signal which was in the fan specs.

The selected power supply is a Mean Well 48 V 60 W compact power supply. It has a wide range of input voltages and low standby power consumption which are shown in Table 1.

Table 1. Mean Well power supply specifications. [3]

Mean Well power supply	Specifications
Output:	
Voltage	48 V DC
Max current	1.25 A
Power	60 W
Plug	5.5 mm barrel
Input:	
Voltage	80-264 V AC
Other:	
Standby	< 0.15 W
Min temp	-30°C
Max temp	70°C

The fans require 48 V DC voltage to operate. It can be powered by bench power supply but for the demo unit that was not optimal. 48 V power supplies are not as common but with some searching the Mean Well 48 V power supply was sourced which worked for this use case. The output power of the power supply is 60 W, which is enough to reliably power the fans yet small enough to fit in the case used.

Batteries were also investigated as a power source. 9 V batteries in series were used to achieve the voltage which was within fan specification. In testing it was discovered the power draw of three fans was too much for the batteries, so a power supply was used to power the fans.

The Arduino is connected to the PC with a USB type B to USB A cable. The Arduino is powered and controlled through a USB connection. The Arduino PWM pin 5 is connected to the fan's PWM lead and this controls the speed of the fans. The fans require 48 V DC and are connected to a power supply. The Arduino is also grounded to the power supply via jumper cable. This is required for the PWM signal to have a reference point. All the pins available on the Arduino can be seen from Figure 5.

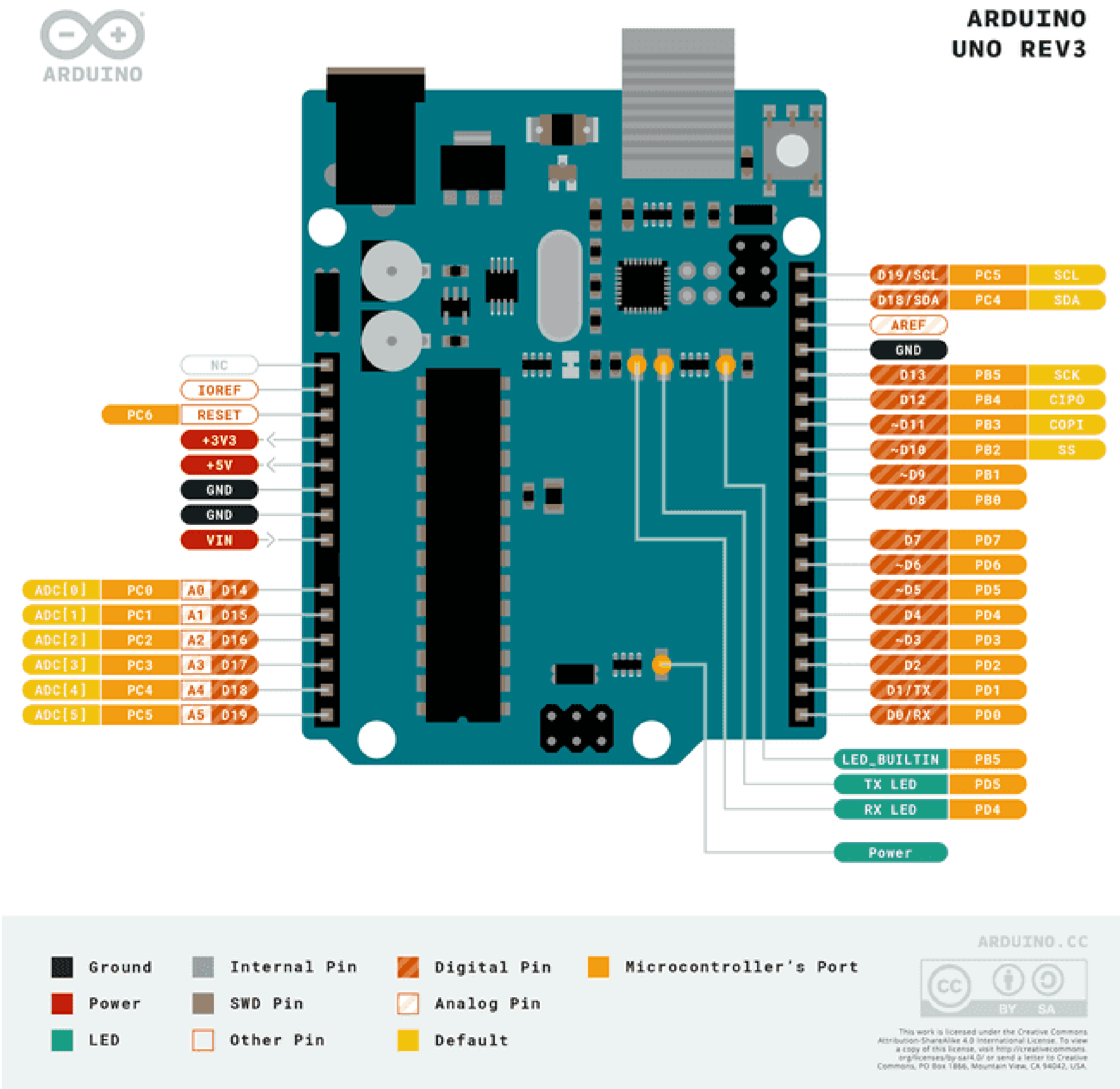

Figure 5. A schematic of Arduino Uno REV3 board and ports. [4]

For the controller pin 5 (D5), ground pin (GND) and USB port were needed for PWM generation. Pin number 5 is selected since it outputs 1kHz PWM signal and this is within specification. Only one port is needed since all the fans must spin at the same speed and the PWM signal is used only to control the separate power connection, so it is not overloaded.

3 CONTROL SOFTWARE

3.1 Software option 1

Windows Remote Arduino Experience is an official program to control an Arduino board running Standard Firmata. The Standard Firmata is a protocol to control the boards inputs and outputs. Main menu of Windows Remote Arduino Experience with the setting for using with USB connection is shown in Figure 6. The protocol can be installed on the Arduino from the Arduino ide software and is not modified for this use case.

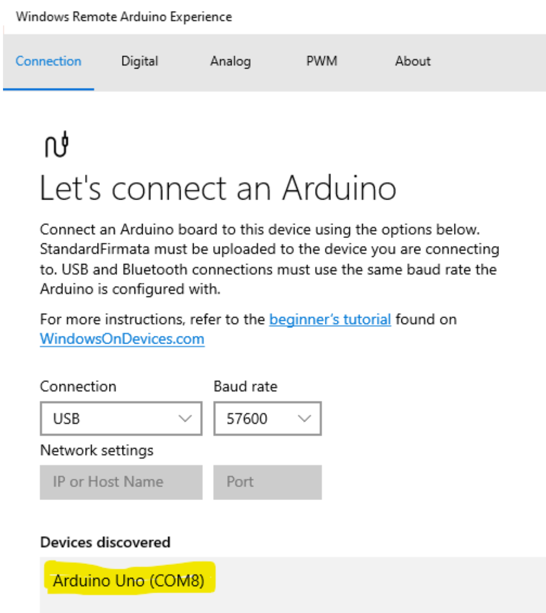

Figure 6. Windows Remote Arduino Experience menu where the device and connection type are chosen [5]

The software requires a USB connection to the PC. After connecting to the PC, the Arduino is selected from the list and the PWM menu from the top is selectable. From the PWM menu each pin can be enabled or disabled. When a pin is enabled, it will provide a PMW signal from that pin. By default, the PWM duty cycle is a 0 but there are 9 levels selectable from 0% duty cycle to 100% duty cycle shown in Table 2. In this fan assembly the control is flipped so a signal with a duty cycle of 0 sets the fans to 100% and 100% duty cycle stops the fans.

Table 2. PWM duty cycle values which are possible with Windows Remote Arduino Experience.

Level (Value)	Duty cycle (%)
0	0
32	12.5
64	25
96	37.5
128	50
160	62.5
192	75
224	87.5
256	100

The windows remote Arduino experience is easy to use and is convenient to set up compared to Python, but it has shortcomings such as less adjustability than the python counterpart. The 12.5% jump in duty cycles is a working way to get a general RPM but is very limiting if a certain speed is required and there is no way to add more levels to it. The available levels and user interface are shown in Figure 7.

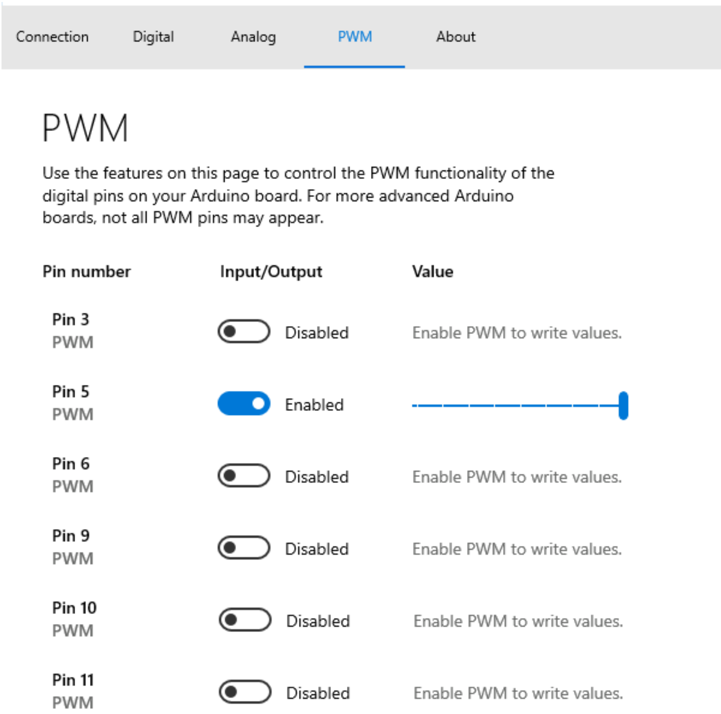

Figure 7. PWM menu of Windows Remote Arduino Experience with pin 5 enabled and value set to 255. [6]

3.2 Software option 2

For the developer software python was used to make the menu design. The menu consists of presets and a developer menu which gives access to more accurate PWM control. The software on the Arduino is Telemetrix. Telemetrix is a protocol used to control the Arduino inputs and outputs from a PC. The board itself has commands which control the inputs and outputs, and the PC is used to send values to the board.

The Telematrix code on the Arduino is not modified but the Python script is written for this use and can be modified when needed. Only the value commands must be configured as in the Telematrix protocol.

Once the Arduino has been connected to the PC and the program has been launched the main menu has four options which are shown in Figure 8. The fan speed is set to zero when the program is launched and will stay there until another value is selected. The first one is fan speed which opens another menu which is visible on the bottom half of Figure 8. The second menu is ambient temperature which contains presets for a certain product which will not be shown here. The developer menu has the possibility to set any value between 0 and 255. This controls the duty cycle of the PWM signal. In both fan speed and developer menu the set fan speed is kept until another one is chosen or when returned to main menu the fan speed is set back to zero. In figure 8 the basic menu structure is represented. In the main menu the exit quits the software and in other menus it returns to the main menu. Figure 9 is a flowchart of the menu used in the Python software.

```
PWM fan controller
*****Type number to select*****
Options:
 1. -- Fan speed
 2. -- Ambient Temperature
 3. -- Developer menu
 4. -- Exit
Select option: 1

Type fan speed:
 1. --   0%     0 rpm
 2. --  30%  1400 rpm
 3. --  50%  2300 rpm
 4. --  70%  3200 rpm
 5. -- 100%  4600 rpm
 6. -- Back
```

Figure 8. Python user interface and some preset values which can be selected.

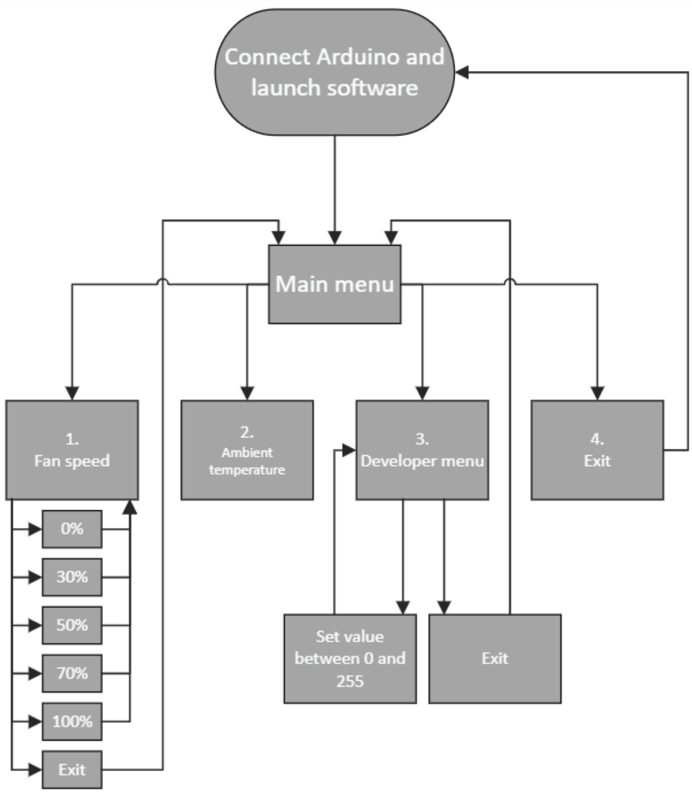

Figure 9. Python software flowchart.

4 TESTING

The control units had to be tested to ensure proper functionality. The control values had to be mapped to find the corresponding fan speeds. The maximum power consumption had to be measured to power the fan assembly properly. Software testing was important, so the user experience is smooth and that all the functions work correctly.

The control values are set between 0 and 255 in both devices. When the value is set to 0 the duty cycle is 100% and at 255 it is 0%, and the fan does not spin.

Arduino software has 9 levels between 0 and 255 with 32 increments. This limited the controllability of the fan but not so much that it could not be used in demonstrations. The different levels were tested and documented with a handheld digital tachometer and verified with multimeter using the frequency setting.

The Python code has a menu with different control options which require measurements of the fan speed at different duty cycles. The Python software has the same range of PWM duty cycle control from 0 to 255 but any value from these can be selected for more accurate control. The measurements could be used to calculate the needed duty cycles that match the presets. These were also measured using the tachometer and verified with multimeter.

The Python developer menu has the option to control the PWM duty-cycle with the control values so the software had to be tested so that only valid values could be used, and errors were handled correctly. Python menu built so that the letter ‘E’ can be used to exit the menus or the last used number. The menu will also reject and display an error message for any other letters or number which are not within limits. For the developer menu the minimum control value was needed to find the value which starts the fans. The minimum PWM duty cycle to start the fans was measured and it is 8.2% and the fan speed is 800 RPM.

For the demo unit a small power supply was needed. The specifications are 48 V and at least 55W. The power need was measured by connecting a multimeter in series on the fan power cable and then the fans were set to maximum speed. The measured maximum current was 1.1A so using Equation (1) the power needed is about 55W. The fans have soft start which makes the fan speed slowly ramp up, so the current does not jump to a higher value.

$$P = U * I \tag{1}$$

Batteries were tested as a power source but were not used in the final design. 9 V batteries were used in series to get a voltage within the fan specification. Both five and six batteries in series were tried since it fit within the fan specification. Six batteries worked better since the voltage of the batteries dropped when under load. Six batteries are 54 V which dropped closer to 48 V under load. With one fan connected to the batteries it worked fine even at full 4600 RPM due to lower power draw. But with three fans the power draw was too high for the batteries and the fan assembly could not reach full speed.

5 DISCUSSION

Overall, the project was successful, the controllers were functional, and the required features were implemented. The demonstration unit was compact and easy to use with easily accessible software. The developer unit had all the features that were requested and the ability to modify the code if needed in the future. Also, the components are easily available, and the device is easily replicated if additional ones are required.

Additional features like live reporting of the current fan speed could be useful as future features but would have taken too long to implement now. An additional quick disconnect for the fan assembly in the development unit could be useful in some installation situations.

6 SUMMARY

In this thesis an Arduino based PWM controller used to control cooling fans was implemented. Two separate versions were built, one for developer use and another to be a mobile unit used for demonstrating fan noise at different speeds. A 60 W compact power supply was chosen for the mobile demonstration unit and the laboratory unit is powered by a bench power supply. Both units are controlled by 5 V PWM signal generated by the Arduino Uno.

The demonstration unit uses Windows Remote Arduino Experience to control the fan speeds. The laboratory unit uses a Python based control software which has more features and more precise control.

In the testing phase limitation with batteries when powering the fan assembly was found out and a suitable 60 W power supply was chosen. The fans speeds were measured and mapped so the Python software could have presets for the required speeds.

7 REFERENCES

[1] J. Huttunen, M. Pärssinen, T. Heikkilä, O. Salmela, J. Manner and E. Pongracz, "Base Station Energy Use in Dense Urban and Suburban Areas," in IEEE Access, vol. 11, pp. 2863-2874, 2023, doi: 10.1109/ACCESS.2023.3234192.

[2] Arduino Uno R3 (Read 15.8.2022). URL: https://docs.arduino.cc/hardware/uno-rev3

[3] Pistorasiamuuntaja (Read 20.8.2022). URL: https://www.elfadistrelec.fi/fi/pistorasiamuuntajat-48v-6a-221w-mean-well-gst220a48-r7b/p/30042762

[4] Arduino UNO Rev3 with Long Pins (Read 15.8.2022). URL: https://docs.arduino.cc/retired/boards/arduino-uno-rev3-with-long-pins

[5] Windows Remote Arduino Experience (Read 25.8.2022). URL: https://apps.microsoft.com/store/detail/windows-remote-arduino-experience/9NBLGGH2041M?hl=fi-en&gl=se&rtc=1