

A Remote Control System for Emergency Ventilators During SARS-CoV-2

Michael Barrow¹, Francesco Restuccia², Mustafa Gobulukoglu, Enrico Rossi³,
and Ryan Kastner¹, *Senior Member, IEEE*

Abstract—As COVID-19 began to grip healthcare systems worldwide, worst-case models predicted huge demands for ventilators. The global community sprang to action, producing a large number of emergency “makeshift” ventilator designs. This brought about another problem: a gap between the quantity of new mechanical ventilators and the number of skilled physicians to operate them. New physicians could not complete training at the pace of ventilator production, which threatened to leave patients sitting untreated, next to unusable ventilators. To address this challenge, we developed a universal remote control system for makeshift ventilators that uses low-cost hardware add-on modules to connect to different ventilators, and a three-tier control architecture to interface the ventilators with telemedicine software. We demonstrate system integration with two representative ventilator designs, adding a remote control option that allows caregivers to quickly and easily monitor and control these ventilators remotely.

Index Terms—Open source hardware, public healthcare, resource management, telemedicine.

I. INTRODUCTION

THE COVID-19 pandemic strained healthcare systems worldwide, resulting in shortages of ventilators, personal protective equipment (PPE), and qualified caregivers [1]. To address the ventilator shortage, many ventilator designs were developed [2]. Designs varied between two extremes.

- 1) The most advanced ventilators contain digital systems to control and monitor ventilator parameters.
- 2) Meet the U.K. NHS RMVS001 specification for emergency therapeutic systems [3].

The most primitive designs are composed of an ambubag that is automatically squeezed by a motorized contraption [4]–[6]. These primitive designs are often built from salvage materials and feature only basic electronic control of respiratory rate.

Ventilated patients must be carefully monitored by a physician to avoid treatment injuries such as barotrauma. This creates a new problem: how can care of ventilated patients

scale given a limited number of caregivers? The scaling problem motivates a need to remotely monitor and control the variety of different makeshift ventilators, so as to bolster available caregiver resources at COVID hot spots, and limit physical access. Remote care subsequently improves access to care in medically underserved and low socioeconomic status (SES) communities and reduces the need for PPE.

Our objective is to develop a universal remote control simple enough to be built from salvaged parts, yet sophisticated enough to be universally compatible with a range of ventilators. The contributions made in this letter include the following.

- 1) Three tiers of remote control architecture compatible with a spectrum of ventilators.
- 2) Hardware add-on designs that facilitate remote access to the major classes of ventilators via a 3.5-mm audio jack.
- 3) Smartphone remote control software that automatically negotiates the most sophisticated tier of control with ventilators featuring our audio jack interface.

We demonstrate the efficacy of our remote control system by integrating it into two different designs that represent the various tiers of remote control architecture, from simple to complex. We describe example PCB hardware add-ons used for the ventilator control. We demonstrate a remote control software that is able to determine and configure itself to control the appropriate tier of ventilator it is connected to. All hardware and software needed to implement the universal remote control system are available along with example telehealth integration in an open-source repository [7].

II. UNIVERSAL VENTILATOR REMOTE CONTROL SYSTEM

A. State of the Art

Prior to the COVID-19 pandemic, remote control of ventilators was limited to a few hospitals that implemented the electronic ICU, or “e-ICU” [8]. Medical devices are controlled using vendor-specific communication mechanisms built on top of Ethernet, WiFi, Bluetooth, etc., that conform to extensive regulation [8], [9]. In the e-ICU model, a physician remotely administers ICU therapeutics, including ventilation, guided by telehealth monitoring of patients in an in-patient setting. Although the e-ICU concept is more than two decades old, prohibitively high equipment costs, system setup complexity, and strict government regulation requirements traditionally left little impetus for remote control operation of ventilators.

However, changes caused by the COVID-19 crisis motivated new designs of remote control ventilators. For example, the U.S. FDA excluded all remote control modifications for ventilators from approval, effectively mandating that the safety risks of remote control were outweighed by medical benefit [10].

Manuscript received July 15, 2021; accepted August 2, 2021. Date of publication August 26, 2021; date of current version February 28, 2022. This work was supported in part by the University of California at San Diego IEM Rapid Response GEM Challenge Program. This manuscript was recommended for publication by H. Tomiyama. (*Corresponding author: Michael Barrow.*)

Michael Barrow, Mustafa Gobulukoglu, and Ryan Kastner are with the Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA 92093 USA (e-mail: mbarrow@eng.ucsd.edu; mgobuluk@eng.ucsd.edu; kastner@eng.ucsd.edu).

Francesco Restuccia is with Scuola Superiore Sant’Anna, 56127 Pisa, Italy, and also with the University of California at San Diego, La Jolla, CA 92093 USA (e-mail: frestuccia@eng.ucsd.edu).

Enrico Rossi is with ELIOS-Tech Srl, 56019 Vecchiano, Italy (e-mail: enrico.rossi@elios-tech.com).

Digital Object Identifier 10.1109/LES.2021.3107837

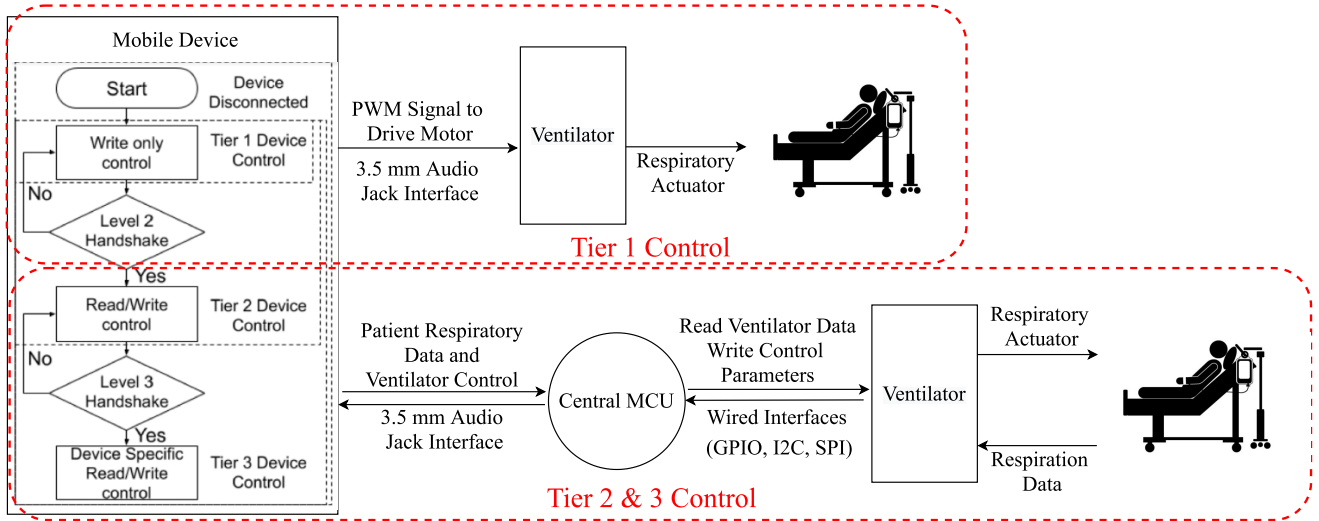


Fig. 1. Three-tier ventilator control architecture. Tier 1 is the most primitive and is intended for dire emergency situations. Tier 2 meets the minimum ventilator specification of “Rapidly Manufactured Ventilator System” RMVS001 for medical use. Tier 3 ventilator control provides additional features to those found in Tier 2 which do not break RMVS001 specification.

With policy relaxed, an urgent need for remote control of ventilators arose. The remote control was needed most acutely in low SES regions, since these areas have fewest doctors per capita, and benefit most from the telehealth aspect of e-ICU.

Unfortunately, the state of the art does not meet the low SES need. For example, although Medtronic developed the “OmniTool” ventilator remote as a result of relaxed FDA mandate [11], it only allows remote control for the Puritan Bennett 9800 Ventilator. This retains cost and complexity issues of traditional e-ICU, while *not* being capable of controlling the proliferation of makeshift ventilators in the COVID-19 crisis setting [2]. As makeshift ventilators would likely be the only option to many low SES patients, existing Bluetooth and WiFi-based methods are also *not* suitable for ventilator control, since most makeshift ventilators are extremely basic, often little more than a motor control circuit. Our goal was to meet the need for a general set of hardware using basic salvageable electronic parts, and software compatible with a broad range of mobile devices. By providing necessary control interfaces for makeshift ventilators to be remotely controlled on a mobile device, ventilators can be integrated into telehealth software, so as to connect low SES patients with remote doctors. We provide an example of this model for two ventilators.

B. Three Tier Control Architecture

Our objective is to enable the use of smartphones as a remote interface to different ventilators. We develop a low-overhead, low-cost three-tier ventilator control architecture that provides remote access to ventilators ranging from very primitive designs to those that meet RMVS001 specification. Our remote control architecture fills the gap between the ventilator and telemedicine applications by providing the hardware and software necessary to gain remote control access to a ventilator. This allows a physician to easily connect to the ventilator, monitor and control patient respiration parameters.

Our architecture allows different ventilators to connect to a telemedicine service, which allows a single physician to remotely monitor and control multiple patient ventilators. Although specifics of the telehealth model are outside the scope of this letter, the idea builds on the supervised ICU

pyramid staffing model [12]. A sample telehealth platform is included in our repository for completeness [7].

The novelty in our approach is the graduating sophistication of ventilator control protocols. By using orthogonal encoding mechanisms, the control of very simple devices is unhindered by the control encoding for very sophisticated devices, and vice versa. Our remote control framework allows plug-and-play smartphone control of devices as simple as a motor-driven pump, up to computer-controlled ICU-grade ventilators.

Our framework has three tiers of makeshift ventilator control, as illustrated in Fig. 1. In a Tier 1 system, a mobile device drives a simplex connection to a ventilator using pulse-width modulation (PWM) to control the speed of a ventilation actuator, which then pushes air into the patient’s lungs. Tier 1 systems control the most basic ventilators found in the literature. These are typically built from salvage and do not use microcontrollers. Therefore, in this mode, the remote cannot read ventilator control parameters or respiratory data. A Tier 2 ventilator control system meets the NHS RMVS001 specification and provides complex control of ventilator parameters with the capacity to read patient data. To meet this standard, we implement an orthogonal encoding using frequency-shift keying (FSK) modulation to support a message passing interface between the mobile device and a central microcontroller unit (MCU). In tier two mode, the mobile device provides a half-duplex connection to a central MCU, which supports a far wider variety of interface connections to the ventilator, such as GPIO, I2C, and SPI. This allows for the read and write of ventilator control registers to control additional parameters, such as tidal volume, positive end-expiratory pressure, peak pressure, and plateau pressure. Finally, the tier three control system utilizes the same architecture as tier two, but allows a ventilator to negotiate additional custom control capability to extend the RMVS001 specification.

Ventilators are connected to a phone via a 3.5-mm audio jack to provide a stable connection with our smartphone remote software. 3.5-mm jacks are found on many devices from different generations. In particular, they are common to lower end phones more prevalent in low SES regions.

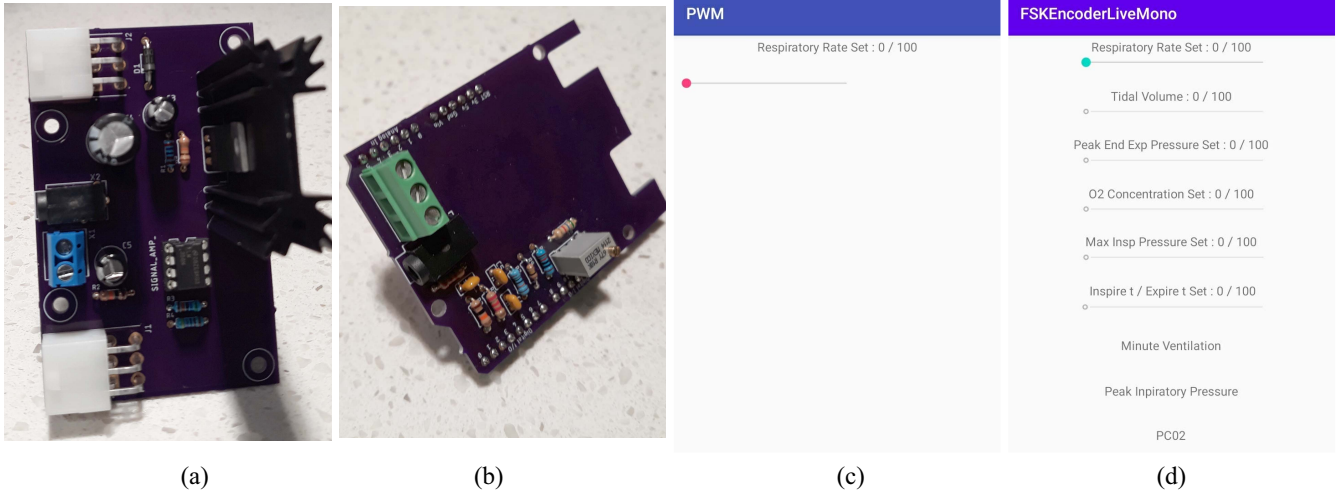


Fig. 2. Examples of ventilator control hardware and software built from plans and code in our repository. (a) is example Tier 1 Hardware; a motor controller compatible with basic ventilators. (b) is example Tier 3 Hardware; a “Shield” compatible with Arduino RMVS001 compliant ventilators. Both (a) and (b) are designed to be built from salvage components on the breadboard in an emergency. (c) and (d) show the remote control app having auto-configured to control Tier 1 and Tier 3 type ventilators, respectively. (c) is presented when the phone is connected to (a), and only respiratory rate can be controlled. (d) is presented when the phone is connected to (b), and all RMVS001 features can be controlled.

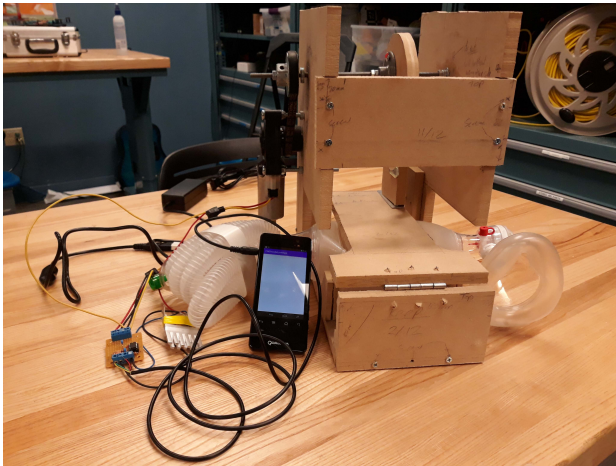


Fig. 3. Example of our Universal Control for a simple Tier 1 makeshift ventilator. Salvage parts from an ATX power supply are used to interface the ventilator with a smartphone via 3.5-mm (audio) jack.

Additionally, audio cable reduces system complexity and build cost.

Fig. 2 shows artifacts needed to implement our system as a field modification to makeshift ventilators. We define two classes of ventilator interface circuit. Fig. 2(a) shows a PCB for simple Tier 1 class ventilators. The PCB is not necessary as the circuit is simple enough to be implemented as emergency field modifications from salvage, for example, by harvesting a commonly found ATX power supply. Fig. 2(b) shows a PCB for Tier 2 and Tier 3 ventilators. This example utilizes an Arduino shield format, which is the most prevalent in Tier 2/3 designs, however, the circuit is a simple differential A/D converter built from passives that can be applied to any MCU with an onboard comparator. Fig. 2(c) and (d) shows our demonstration application negotiating Tier 1 and Tier 3 control, respectively. All designs and code featured in Fig. 2 are provided online [7].

In the following sections, we demonstrate applying our control method as a field modification to two ventilator systems.

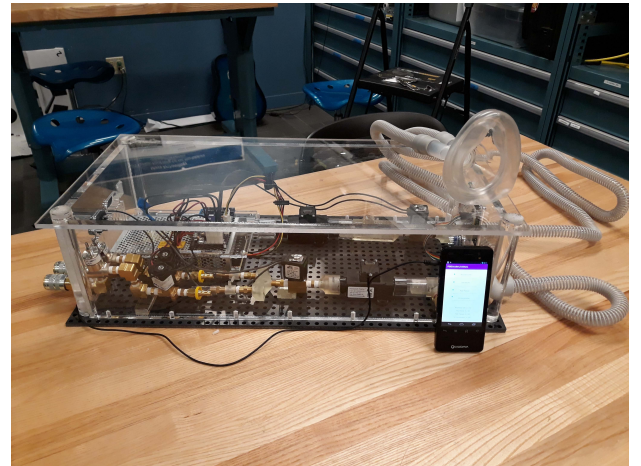


Fig. 4. Example of our Universal Control for an Arduino UNO-based Tier 2 RMVS001 compliant makeshift ventilator. Our Arduino shield is used to interface a smartphone to the ventilator via the 3.5-mm audio jack.

C. Example Tier 1 Ventilator With Our Remote Control

Fig. 3 shows a prototype of our Tier 1 hardware interface which uses salvaged electronic components from an ATX power supply to operate a makeshift ventilator based on a car window motor. The ventilator is approved for emergency use during the COVID pandemic by the Spanish government [2]. The smartphone audio jack is connected to the hardware interface using Tier 1 control to vary the respiratory rate.

D. Example Tier 2 Ventilator With Our Remote Control

Fig. 4 shows an Arduino UNO-based RMVS001 compliant makeshift ventilator design interfaced with our Tier 2 remote control system. Adding remote control via our system is accomplished simply by plugging in the “shield” form factor PCB shown in Fig. 2(b) and connecting a smartphone running our control software via a 3.5-mm jack cable. The ventilator in Fig. 4 is designed for hospital use with main line compressed gas intakes at 50–120 PSI. It is a hybrid pressure/volume-controlled model, with an automated

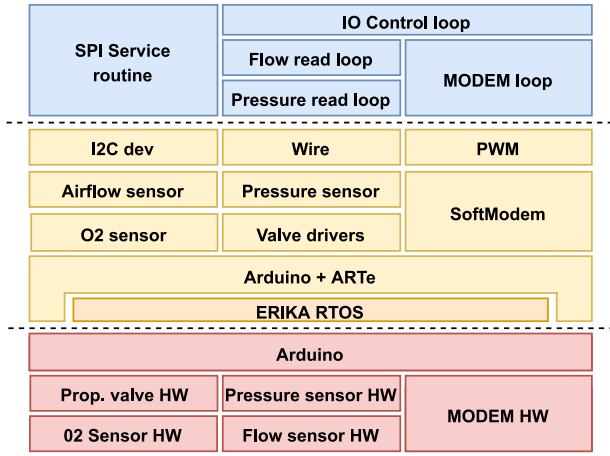


Fig. 5. Software architecture of the Tier 2 makeshift ventilator example. The minimal safety-critical requirements are provided by the ARTe framework.

O₂ mixer system controlled by O₂ exchange monitoring of the patients lungs. It features over/under pressure valves for unsupported breathing in a power failure event.

By providing a stackable Arduino shield design, our remote interface can be integrated into many existing Arduino-based ventilators [13]. Our solution is, therefore, plug and play with most Arduino-based makeshift ventilator designs.

We designed the remote interface to be simple enough to be implemented on a breadboard using a basic hobby electronic kit and a pair of headphones. This is helpful in emergency conditions where PCB lithography tools are not available.

Software Architecture: The popularity and accessibility of the Arduino platforms make them great candidates for provisioning and development in emergency conditions. However, certain challenges must be addressed when Arduino is leveraged for the deployment of safety-critical applications. Specifically, ventilation therapy requires safety-critical control of a pneumatic system that drives the breathing cycle of a sedated patient using feedback from an array of pressure and airflow sensors. In the Tier 2 example ventilator, required timing predictability in execution is obtained by leveraging the ARTe framework. ARTe provides an extended programming model enabling the development of multithread real-time applications by leveraging the ERIKA enterprise [14] real-time operating system (RTOS) and executing on standard Arduino platforms. With ARTe, a myriad of software drivers provided by the Arduino community are available to connect peripherals. This dramatically reduces the time needed to adapt to part shortages and facilitates rapid adoption of a hardware configuration that meets performance requirements, safety specification, and part availability. The software stack of the example design is shown in Fig. 5. The application is structured in multiple ARTe loops in blue. ARTe loops incorporate Arduino libraries, and community-provided libraries to interface with the valves and sensors of the ventilator in yellow. Access to the hardware resources in red from the ARTe loops is managed by the ERIKA RTOS. Each critical functionality is implemented in a separated ARTe loop (i.e., IO control, pressure monitoring, gas flow, and our remote control drivers each have independent loops). Each ARTe loop is associated with a time period, corresponding to its deadline and translated into an ERIKA task by the ARTe framework. The tasks are scheduled by the ERIKA RTOS

scheduler, which guarantees the timing requirements of the tasks (i.e., each task completes the execution within the corresponding deadline) and that the execution of the critical functionalities is isolated from each other. Interloop communication is marshaled via ARTe data structures, which eliminate race conditions with an automatic transparent double buffer system [15].

III. CONCLUSION

The COVID-19 pandemic caused a large surge in the demand for ventilators. This was responded to via the design of makeshift ventilators. However, this only gave rise to another problem: How to control these ventilators given a limited amount of physicians? We addressed this problem by developing a three-tier universal ventilator remote control architecture. We demonstrate the architecture by integrating remote control into two makeshift ventilators at different ends of design complexity found in the literature. By integrating telehealth with our architecture, our remote control scales the number of patients a single physician can treat.

REFERENCES

- [1] M. L. Ranney, V. Griffith, and A. K. Jha, "Critical supply shortages—The need for ventilators and personal protective equipment during the Covid-19 pandemic," *New England J. Med.*, vol. 382, no. 18, p. e41, 2020.
- [2] J. M. Pearce, "A review of open source ventilators for COVID-19 and future pandemics," *F1000Research*, vol. 9, p. 218, Mar. 2020.
- [3] MHRA. (Mar. 2020). *Rapidly Manufactured Ventilator System (RMVS)*. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/879382/RMVS001_v4.pdf
- [4] A. H. Kwon, A. H. Slocum, Jr., D. Varelmann, and C. G. S. Nabzyk, "Rapidly scalable mechanical ventilator for the COVID-19 pandemic," *Intensive Care Med.*, vol. 46, no. 8, pp. 1642–1644, 2020.
- [5] M. Palmer. (Mar. 2020). *Turning Windscreen Wiper Motors Into Emergency Ventilators*. [Online]. Available: <https://sifted.eu/articles/coronavirus-windscreen-ventilator/>
- [6] A. Vasan *et al.*, "MADVent: A low-cost ventilator for patients with COVID-19," *Med. Devices Sens.*, vol. 3, no. 4, 2020, Art. no. e10106.
- [7] (Jul. 2021). *TVCV19 Official Github Page*. [Online]. Available: <https://github.com/KastnerRG/TVCV19>
- [8] C. Becker, "Remote control. specialists are running intensive-care units from remote sites via computers, and at least one health system with the e-ICU is reaping financial rewards—and saving lives," *Mod. Healthc.*, vol. 32, no. 8, pp. 40–46, 2002.
- [9] P. Raghavan and T. Morgan, "Wireless medical room control arrangement for control of a plurality of medical devices," U.S. Patent 9 740 826, Aug. 22, 2017.
- [10] *Enforcement Policy for Ventilators and Accessories and Other Respiratory Devices During the Coronavirus Disease 2019 (COVID-19) Public Health Emergency: Guidance for Industry and Food and Drug Administration Staff*, United States Food Drug Admin., Silver Spring, MD, USA, 2020.
- [11] Medtronic. (Jul. 2021). *Remote Access Enablement, Puritan Bennett 9800 Ventilator Installation and Reference Guide*. [Online]. Available: <https://www.medtronic.com/content/dam/covidien/library/us/en/product/acute-care-ventilation/puritan-bennett-980-remote-access-reference-guide.pdf>
- [12] D. Kumaraiah, N. Yip, N. Ivaseu, and L. Hill, "Innovative ICU physician care models: Covid-19 pandemic at newyork-presbyterian," *NEJM Catalyst Innovat. Care Del.*, vol. 1, no. 2, 2020.
- [13] D. Cuartielles *et al.* (Apr. 2021). *Combating COVID-19 Conference: Arduino-Based Ventilators*. [Online]. Available: https://www.youtube.com/watch?v=Tt1ouUcUzM&ab_channel=Arduino
- [14] (Jul. 2021). *Erika Enterprise RTOS Kernel*. [Online]. Available: <http://erika.tuxfamily.org/drupal/>
- [15] (Jul. 2021). *The Official ARTe Website*. [Online]. Available: arte.retis.santannapisa.it