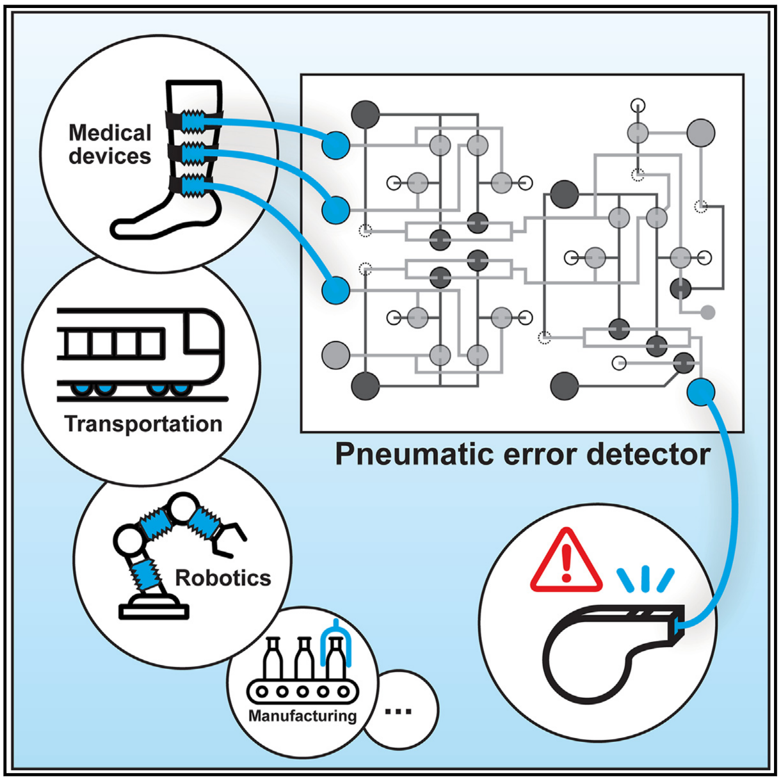
Air-powered logic circuits for error detection in pneumatic systems

Graphical abstract



Highlights

- Air-powered logic circuits can detect problems in pneumatic systems
- The pneumatic error detector calculates parity bits using microfluidic valves
- The device successfully detects failures in a wearable medical device
- A low-cost (<$1 USD) way to detect problems using no electronic sensors

Authors

Shane Hoang, Mabel Shehada, Zinal Patel, Minh-Huy Tran, Konstantinos Karydis, Philip Brisk, William H. Grover

Correspondence

wgrover@engr.ucr.edu

In brief

An air-powered error detector is developed for identifying problems in pneumatic systems. The device contains microfluidic valves that act like transistors in a pneumatic logic circuit, and it performs parity bit calculations to detect problems such as air leaks or blockages. In our demonstration, the pneumatic error detector monitors a wearable medical device and alerts the wearer of an error by sounding a whistle.

4 **Validate**
Functional device with real-world testing, ready to scale

Hoang et al., 2024, Device 2, 100507
November 15, 2024 © 2024 The Author(s).
Published by Elsevier Inc.
https://doi.org/10.1016/j.device.2024.100507

Device

CellPress
OPEN ACCESS

Article

Air-powered logic circuits for error detection in pneumatic systems

Shane Hoang,[1] Mabel Shehada,[1] Zinal Patel,[1] Minh-Huy Tran,[1] Konstantinos Karydis,[2] Philip Brisk,[3] and William H. Grover[1,4,*]

[1]Department of Bioengineering, University of California, Riverside, 900 University Avenue, Riverside, CA 92521, USA
[2]Department of Electrical and Computer Engineering, University of California, Riverside, 900 University Avenue, Riverside, CA 92521, USA
[3]Department of Computer Science and Engineering, University of California, Riverside, 900 University Avenue, Riverside, CA 92521, USA
[4]Lead contact
*Correspondence: wgrover@engr.ucr.edu

https://doi.org/10.1016/j.device.2024.100507

THE BIGGER PICTURE Air-powered pneumatic systems are used in a wide variety of mechanical systems ranging from train brakes to assembly line robots, breast pumps, and medical ventilators. Sensors can be added to these systems to detect failures, but this additional electronic hardware adds cost, complexity, and peripheral safety concerns to the system. In this work, we present an inexpensive and easy-to-manufacture air-powered logic device that can detect and respond to problems in pneumatic systems without using electronic sensors. This work shows that pneumatic logic can be used to make a wide range of important air-powered systems safer and less expensive.

SUMMARY

Pneumatic control systems are common in manufacturing, healthcare, transportation, robotics, and many other fields. Undetected failures in pneumatic systems can have serious consequences. In this work, we present an air-powered error detector that can identify failures in pneumatic systems. This device contains a pneumatic logic circuit of 21 microfluidic valves that calculates the parity bit corresponding to several pneumatic control bits. If a problem such as an air leak or blockage occurs, then the calculated and expected parity bits will not match, and the device outputs an error signal to alert the user or to shut down the system. As a proof of concept, we used the device to detect anomalies in an intermittent pneumatic compression (IPC) medical device. By providing a simple and low-cost way to detect problems without using sensors, the pneumatic error detector can promote safety and reliability across a wide range of pneumatic systems.

INTRODUCTION

Air-powered systems play important roles in health care, manufacturing, transportation, robotics, and many other areas. When these pneumatic systems fail, the consequences can be catastrophic. Consequently, it is desirable to endow these systems with error-detection strategies that can detect failures in the pneumatic actuation system and take appropriate action. Current error-detection strategies employ electronic sensors that monitor air pressure or flow rate at various points in a system and relay this information to a separate control system, often a computer or microcontroller, for analysis and error mitigation. This electronic monitoring hardware adds considerable complexity, size, and cost to the overall system. This approach is also problematic in soft robotic systems, which use pneumatics to control air-filled actuators[1–8] and are sensitive to size, weight, and power (SWaP) considerations. Considering that each independent actuator typically already has a separate pneumatic control line, adding yet another set of components for error detection further impacts SWaP efficiency and defeats many of the advantages of soft robotics (their simplicity, autonomy, low cost, biomimetic design, and having few or no electronic components).

In this work, we present a pneumatic logic system for detecting errors in pneumatic systems. Pneumatic logic traces its origins to century-old air-powered systems for climate control in large buildings[9] and self-playing pianos,[10] it reached a pinnacle in 1964 with the FLODAC computer[11] but fell out of favor when electronic transistors and integrated circuits became ubiquitous in computing hardware. Almost two decades ago, researchers began using microfluidic valves in logic circuits to control "lab-on-a-chip" devices,[12–14] and this is still an active research area.[15,16] Pneumatic logic has also attracted interest as a potential low-cost and non-electronic method for controlling

Please cite this article in press as: Hoang et al., Air-powered logic circuits for error detection in pneumatic systems, Device (2024), https://doi.org/10.1016/j.device.2024.100507

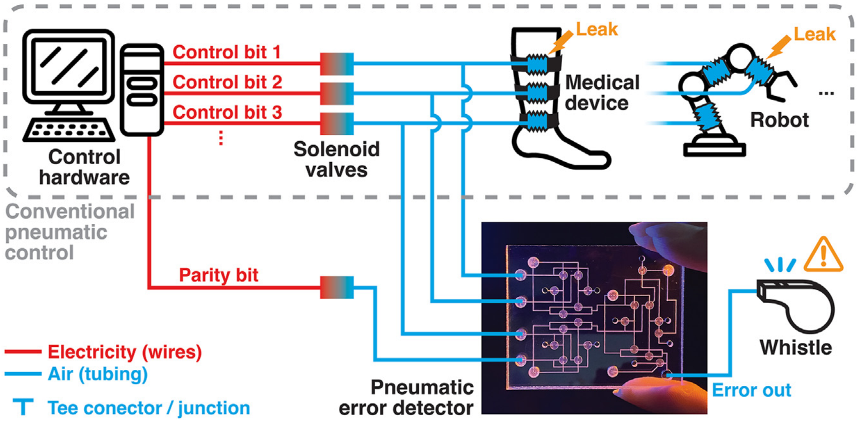


Figure 1. Using the pneumatic error detector to detect problems during the operation of a typical pneumatically actuated system
In this example of a conventional pneumatic control system (gray dotted line), electronic hardware controls the states (**1** or **0**) of three binary control bits, three solenoid valves convert these bits to pneumatic signals (vacuum for **1** and atmospheric pressure for **0**), and the pneumatic signals are connected to the system being controlled (e.g., medical device, robot) using tubing. To add the pneumatic error detector to this control system, the program on the electronic control hardware is modified to calculate the parity bit corresponding to the values of the three control bits, a fourth solenoid valve converts this parity bit to a pneumatic signal (again using vacuum for **1** and atmospheric pressure for **0**), and the three control bits and one expected parity bit are connected to the pneumatic error detector using tubing. The pneumatic error detector uses an air-powered logic circuit consisting of 21 monolithic membrane valves to repeat the parity bit calculation and compare the result to the expected parity bit value. If the two values for the parity bit are different, then this indicates that one of the control bit signals is incorrect due to, for example, a leak occurring in the medical device or soft robot, and the error detector responds by automatically outputting **1** (vacuum) on an error line. This pneumatic error signal can be used to alert the operator (using a whistle here), initiate a system shutdown, or take some other corrective action.

air-powered systems such as soft robots,[17–23] biomedical devices,[22,24] and haptic wearables.[25] To build the air-powered error detector, we used monolithic membrane valves,[26] a microfluidic valving technology with a long history of use in complex pneumatic logic devices.[12,13,15,27–34]

Parity bits for error detection

There are many different methods for error detection in computing and communication systems. In this work, we used parity bits for error detection; this fundamental yet effective error detection technique has been used in electronic computing since at least the early 1950s.[35] In parity bit-based error detection, the current values (**1** or **0**) of several binary bits are used to calculate the value of a parity bit. For example, consider three binary bits with the values **1**, **1**, **0**. To calculate the parity bit that corresponds to the values of these three bits, we can use any one of these (mathematically equivalent) methods:

- Calculate the Boolean exclusive OR (or XOR) of all the bits: **1** XOR **1** XOR **0** = **0**.
- Calculate the sum of the bits *modulo* 2: **1** + **1** + **0** = 2 (mod 2) = **0**.
- Count the number of 1s in the values of the bits; the parity bit is **1** if the count is an odd number and **0** if the count is an even number. Since there are two 1s in **1**, **1**, **0** and two is even, the parity bit is **0**.

This expected parity bit value is then transmitted along with the original bits to some recipient, and the recipient repeats the parity bit calculation using the values of the bits they received. If the values of the bits were unchanged during transmission, then the value of the parity bit will also be unchanged, and the recipient can be confident that no single-bit errors occurred during transmission. However, if a single bit changed state during transmission (e.g., if **1**, **1**, **0** was received as **1**, **0**, **0**), then the parity bit calculated by the recipient would also change (in this case, from **0** to **1**) and would no longer match the expected parity bit. The recipient would know that an error has occurred and that one of the received bit values is wrong.

RESULTS

The pneumatic logic error detector uses air flowing through a network of 21 monolithic membrane valves to calculate the value of a parity bit corresponding to the states of three pneumatic control signals. If the calculated and expected parity bits differ at any point, then an error has been detected (one of the control signals is in the wrong state). When this happens, the pneumatic error detector automatically outputs a pneumatic signal that can be used to alert a user, shut down the system, or take other action. As a proof of concept, we used the pneumatic error detector to automatically detect different types of failures in the operation of an important medical device, an intermittent pneumatic compression (IPC) device that prevents the formation of life-threatening blood clots in the wearer's legs.[36–38] When a failure occurs (e.g., air leak, blocked air line) that would compromise the efficacy of the IPC device and possibly endanger the wearer of the device, the pneumatic error detector senses this error and, in this demonstration, alerts the wearer or nearby healthcare professionals by blowing a whistle. This pneumatic error detector is a direct and low-cost way to add error detection to a wide variety of pneumatic-controlled systems.

Figure 1 provides an overview of using the pneumatic error detector to sense problems with the operation of a pneumatically actuated system. The top portion of Figure 1 represents a typical system for controlling pneumatic devices: electronic control hardware operates solenoid valves, which in turn supply air pressure or vacuum to the device being controlled. More specifically, a program running on the electronic hardware controls the states—either **1** (True) or **0** (False)—of each of several control bits (in this example, control bits 1, 2, and 3 in Figure 1). The solenoid valves effectively convert the control bit values to pneumatic signals: a control bit value of **1** produces a vacuum on

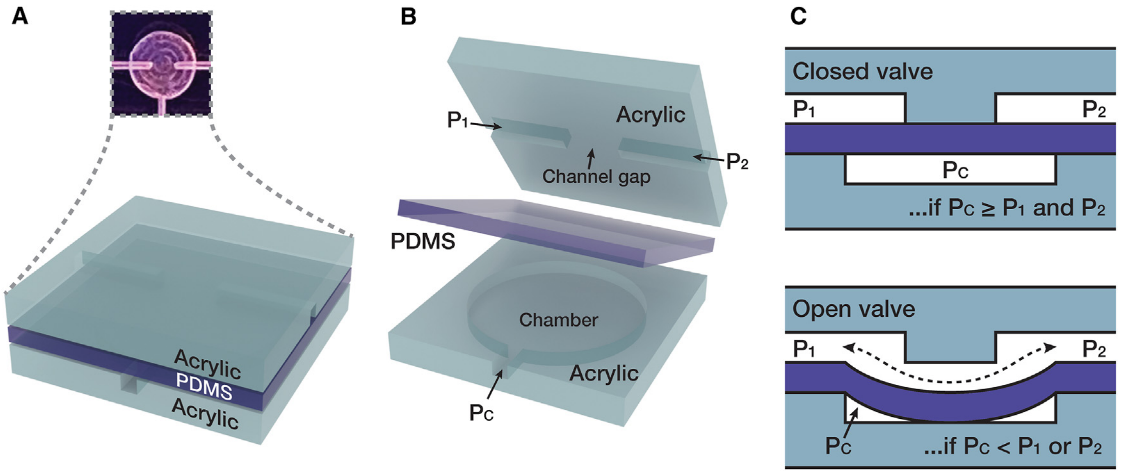


Figure 2. Design of monolithic membrane valves
Top (A), exploded (B), and cross-sectional (C) views of a monolithic membrane valve.

that control line, and a value of **0** produces atmospheric pressure. Finally, those pneumatic signals are connected via tubing to the system being controlled, such a medical device[36–38] or robot.[1–8] This setup is representative of many pneumatic control systems that switch multiple independent pneumatic control lines between two different pressure states.

To add the pneumatic error detector to the conventional control system shown at the top of Figure 1, a few small changes are necessary. First, the computer program running on the electronic control hardware is modified to calculate the value of the parity bit that corresponds to the values of control bits 1, 2, and 3 at each step during the device operation sequence. Second, an additional solenoid valve is used to convert the parity bit calculated by the electronic hardware into its pneumatic representation, again using vacuum for **1** and atmospheric pressure for **0**. Third, the pneumatic error detector device is connected to the pneumatic control bits 1, 2, and 3 and the pneumatic parity bit using tubing. The pneumatic control bit connections are made using tee junctions so that the pneumatic control signals reach both the pneumatic error detector and the system being controlled. Finally, five additional solenoid valves not shown in Figure 1 are used to provide the vacuum and the atmospheric pressure needed to operate and reset the error detector (details in experimental procedures below).

Once connected to the pneumatic control system as shown in Figure 1, the pneumatic error detector uses monolithic membrane valves and flowing air to repeat the parity bit calculation originally performed by the electronic control hardware and compares the resulting parity bit value to the one calculated by the electronic hardware. If the two parity bits agree (both are **0** or both are **1**), then the values for control bits 1, 2, and 3 have passed successfully from the computer to the system being controlled—no error has occurred, and the error detector outputs **0** (atmospheric pressure). However, if the two parity bits disagree (one is **0** and the other is **1**), then one of the pneumatic signals is different from what the computer intended—an error has occurred. The error detector outputs **1** (vacuum), which, in this example, causes a whistle alarm to sound, alerting those nearby that an error has occurred. In this manner, the pneumatic error detector can give pneumatically controlled systems the ability to detect and respond to errors without the need for electronic sensing hardware.

Pneumatic error detector design and operation

The pneumatic error detector consists of three layers: a featureless polydimethylsiloxane (PDMS) silicone rubber membrane sandwiched between two engraved acrylic plastic sheets. Monolithic membrane valves[26] are formed wherever a gap in an engraved channel in one acrylic layer is located directly across the PDMS membrane from an engraved chamber in the other acrylic layer, as shown in Figures 2A and 2B. A cross-section through a valve (Figure 2C) shows that these valves are normally closed; the PDMS membrane normally rests against the channel gap and stops air from flowing across the gap. When a vacuum is applied to the chamber, the PDMS membrane is pulled into the chamber and away from the channel gap; this creates a path for air to flow across the gap and the valve opens. More generally, for a valve with pressures P_1 and P_2 at the two connections to the valved channel and pressure P_C at the chamber:

- If $P_C \geq P_1$ and $P_C \geq P_2$, then the valve will be closed.
- If $P_C < P_1$ or $P_C < P_2$, then the valve will be open; air will flow from channel 1 to channel 2 as long as $P_1 > P_2$, or from channel 2 to channel 1 as long as $P_2 > P_1$.

Full details on how specific pressures and valve designs affect monolithic membrane valve behavior are available elsewhere.[12,13,26]

Multiple monolithic membrane valves can be connected to form more complex pneumatic logic gates. For example, two valves in series form a Boolean AND gate as shown in Figure 3A: air can flow through the valves only if both valves receive vacuum (i.e., if bit 1 = **1** AND bit 2 = **1**). Likewise, two valves in parallel form a Boolean OR gate shown in Figure 3B: air can flow through the gate if either valve (or both) receives vacuum (i.e., if bit 1 = **1** OR

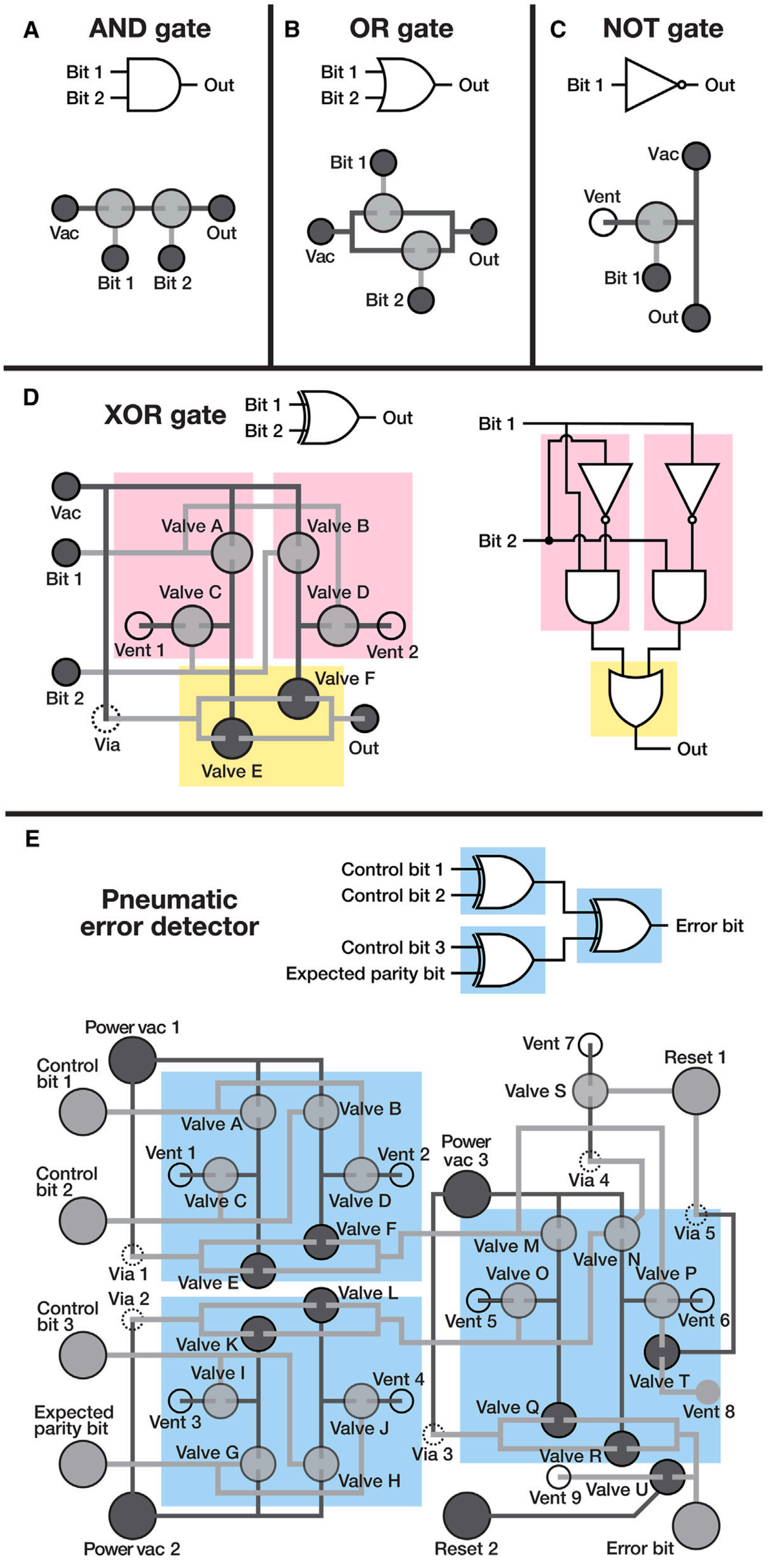


Figure 3. Design of the pneumatic error detector
Valve-based pneumatic logic circuits include the Boolean AND (A), OR (B), NOT (C), and XOR (D). The pneumatic error detector (E) uses three XOR gates to calculate the parity bit corresponding to the values of three control bits. If the result differs from the expected parity bit value, then the detector outputs a vacuum on the error bit.

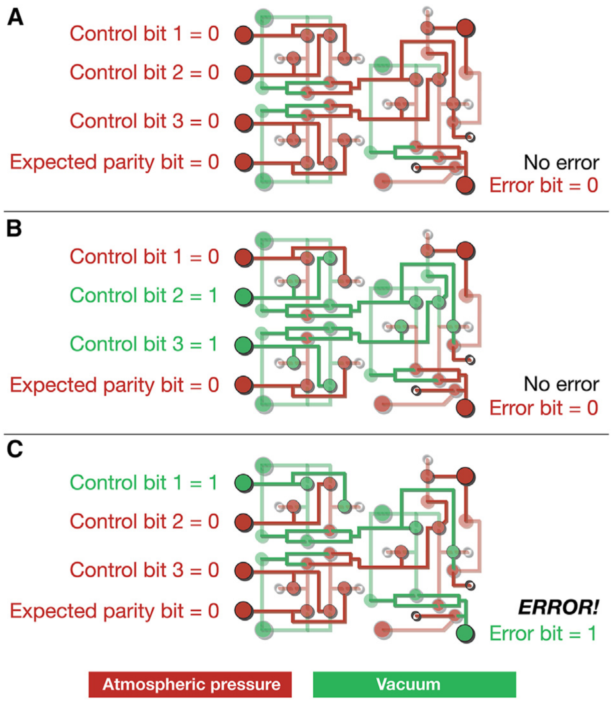


Figure 4. Pressures inside the pneumatic error detector's channels during three example calculations

Channels under atmospheric pressure are colored red, and channels under vacuum are green. In examples (A) and (B), the error detector confirms that the expected and calculated parity bits match, so no error is detected and the error output remains at atmospheric pressure (**0**). In example (C), the expected parity bit of **0** does not match the calculated parity bit of **1**, so the error detector outputs a vacuum (**1**), indicating a problem has been detected.

bit 1 = **1**). A Boolean NOT gate (Figure 3C) is formed by connecting a vent (a drilled hole to the atmosphere) to a vacuum supply. When the valve receives vacuum, the valve opens and atmospheric pressure air reaches the output of the gate, but when the valve receives atmospheric pressure, the valve remains closed and vacuum reaches the output. In this manner, the output is always the opposite of the input (in other words, if bit 1 = **1**, then the output is **0**, and if bit 1 = **0**, then the output is **1**) as expected with a Boolean NOT or negation operator.

These basic pneumatic logic gates can in turn be combined to create more complex logic circuits. Figure 3D shows a gate that is essential to the operation of the error detector, an XOR gate. The gate consists of two sets of AND and NOT gates (highlighted in pink) and an OR gate (highlighted in yellow). When bit 1 and bit 2 are both **0** (atmospheric pressure), valves A, B, C, and D remain closed, and valves E and F remain at atmospheric pressure (closed); consequently the output of the gate remains at atmospheric pressure (**0**). Similarly, when bit 1 and bit 2 are both **1** (vacuum), valves A, B, C, and D open, vacuum is vented through vents 1 and 2, and valves E and F remain at atmospheric pressure (closed) and the output of the gate again remains at atmospheric pressure (**0**). However, when bit 1 and bit 2 are different (e.g., bit 1 is **1** or vacuum and bit 2 is **0** or pressure), vacuum reaches one of the inputs of the OR gate (in this case, valve A receives vacuum and opens while valve C remains at atmospheric pressure and stays closed, and vacuum reaches valve E, which opens); this in turn allows vacuum to reach the output of the gate (**1**). In this manner, the output of the XOR gate is **1** if bit 1 and bit 2 are different, and **0** if bit 1 and bit 2 are the same, as expected with an XOR gate. Details on this and other valve-based pneumatic logic gates are available elsewhere.[13]

Finally, three XOR gates (each highlighted in blue) are combined together to form the pneumatic error detector shown in Figure 3E. This pneumatic circuit comprises 21 valves; 18 of the valves are used in 3 XOR gates (valves A–F, G–L, and M–R) and 3 additional valves (S, T, and U) are used to vent trapped vacuums to reset the device between operations. The device has three control bit inputs (bits 1, 2, and 3), one expected parity bit input, one error bit output, three "power vacuum" inputs that receive vacuum to power the device, and two "reset" inputs that are used to open valves S, T, and U to vent trapped vacuums. The device also contains five "vias" (holes punched through the PDMS membrane prior to device assembly) to allow pneumatic signals to pass from one layer to another, and nine drilled vents to admit atmospheric-pressure air into the device. Figure 4 depicts the contents (vacuum or atmospheric pressure) of every feature inside the device during three sample computations. Table 1 shows the expected value of the error bit output for each of the 16 different combinations of values for control bit 1, control bit 2, control bit 3, and expected parity bit. The values in the top half of the table correspond to "correct" expected parity bits (no errors detected), and the values in the bottom half correspond to "incorrect" parity bits (errors detected).

Testing the pneumatic error detector

To test the operation of the pneumatic error detector, we operated the device using all 16 possible combinations of **1**s (vacuum) and **0**s (atmospheric pressure) to the three control bit inputs and one expected parity bit input while measuring the

Table 1. Truth table for the pneumatic error detector

Control bit 1	Control bit 2	Control bit 3	Expected parity bit	Error bit
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	0
0	0	1	1	0
1	0	1	0	0
0	1	1	0	0
1	1	1	1	0
0	0	0	1	1
1	0	0	0	1
0	1	0	0	1
1	1	0	1	1
0	0	1	0	1
1	0	1	1	1
0	1	1	1	1
1	1	1	0	1

Expected values of the error bit for all 16 different combinations of values for the control bits and expected parity bit.

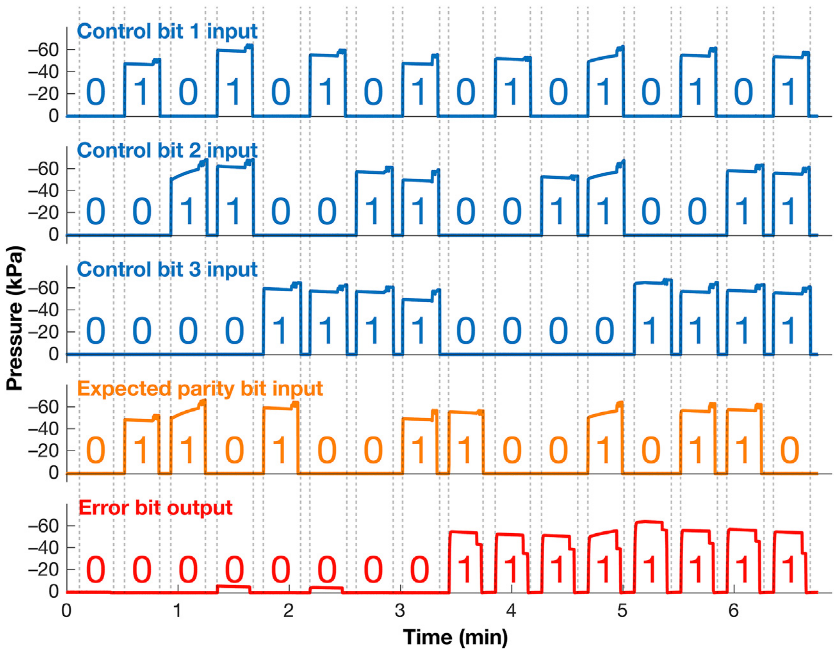


Figure 5. Characterizing the performance of the pneumatic error detector

Pressure measured at each of the three control bit inputs (blue), one expected parity bit input (orange), and one error output (red) while applying all 16 possible combinations of **1**s (vacuum) and **0**s (atmospheric pressure) to the control bit inputs and parity bit input in the order shown in Table 1. During the first eight combinations (times from 0 to 3.5 min), the expected parity bit is correct or consistent with the values of the three control bits, and the near-zero (atmospheric) pressures measured at the error output confirm that no error has occurred. However, during the last eight combinations (times from 3.5 to 7 min), the expected parity bit is intentionally incorrect (the opposite of what it should be), and the vacuums measured at the error output confirm that the device has successfully detected these errors. Results from 15 successful replicates of this experiment (some with different timings) are available in Figures S1, S2, and S3.

pressure at the error output. The input combinations were tested in the order shown in Table 1, and each combination was maintained for 15 s. Figure 5 shows the pressure measured at each of the four inputs and one output during a typical experiment. The **1**s and **0**s are superimposed on the pressure measurements to indicate whether the measured pressure corresponds to **1** (vacuum) or **0** (atmospheric pressure). In the left half of Figure 5 (corresponding to the top half of Table 1), the value of the expected parity bit input is intentionally always consistent with the provided values for the control bit inputs, and the pressure measured at the error output remains at or close to atmospheric pressure (**0**), indicating that no error has occurred. However, in the right half of Figure 5 (corresponding to the bottom half of Table 1), the value of the expected parity bit is intentionally always incorrect, and the pressure measured at the error output always goes to vacuum, showing that each simulated error has been successfully detected. The pattern of **1**s and **0**s measured in Figure 5 matches the expected pattern in Table 1, thereby confirming that the pneumatic error detector functions correctly.

We repeated the experiment shown in Figure 5 for a total of five runs. We also repeated the experiment using shorter wait times after each input combination (10 s and 5 s; five repeats of each). All 15 of these runs yielded correct values for all bits, with minimal differences between the measured pressures across the runs (data provided in Figures S1, S2, and S3).

Detecting errors in a model medical device

To validate the pneumatic error detector in a real-world application, we used it to monitor a model medical device, an IPC device commonly used to prevent the formation of blood clots in a wearer's legs. Our model IPC device, shown in Figure 6A, consists of three flexible plastic bellows connected via 3D-printed buckles to nylon straps that wrap around a simulated leg. When vacuum is applied to one of the IPC device's bellows, the bellows contracts and squeezes the corresponding region on the simulated leg. A software program written in our valve control language OCW[39] sets the values of the three control bits, which in turn control the three solenoid valves that apply vacuum (when the control bit is **1**) or atmospheric pressure (when the control bit is **0**) to the three IPC bellows. The program contracts the bellows one at a time in sequence: first setting control bit 1 to **1** (shown in Figure 6B; the contracted bellows is indicated using a white dotted line), then setting control bit 2 to **1** (Figure 6C), and then setting control bit 3 to **1** (Figure 6D). This pattern repeats (Figures 6B → 6C → 6D → 6B → 6C → 6D …), creating a peristaltic squeezing motion that is meant to encourage blood flow in the leg. The same computer program also calculates the value of the parity bit corresponding to the values of the three control bits at each step in the actuation pattern, and an additional solenoid valve outputs the pneumatic version of this expected parity bit (**1** = vacuum and **0** = atmospheric pressure) whenever error checking is desired. The computer program also controls three solenoid valves that provide vacuum to power the error detector, and two solenoid valves that reset the error detector after operation. An additional free bellows (labeled "Expected parity bit" in Figure 6) was added to the expected parity bit pneumatic line so that the state of this line can be visualized during operation (contracted bellows = **1** and extended bellows = **0**). The three pneumatic control bit signals and one pneumatic expected parity bit signal are connected to the pneumatic error detector, which repeats the parity bit calculation on the three control signals and compares the result to the expected parity bit. If the two values are not the same, then the error detector sets its error output to **1** (vacuum).

In this demonstration, our aim was for the pneumatic error detector to alert the wearer by blowing a whistle when an error is detected. Since most whistles use positive pressure (not vacuum) to generate a sound, we needed a simple method for converting the vacuum at the error output to a positive pressure for powering the whistle. We accomplished this by using a

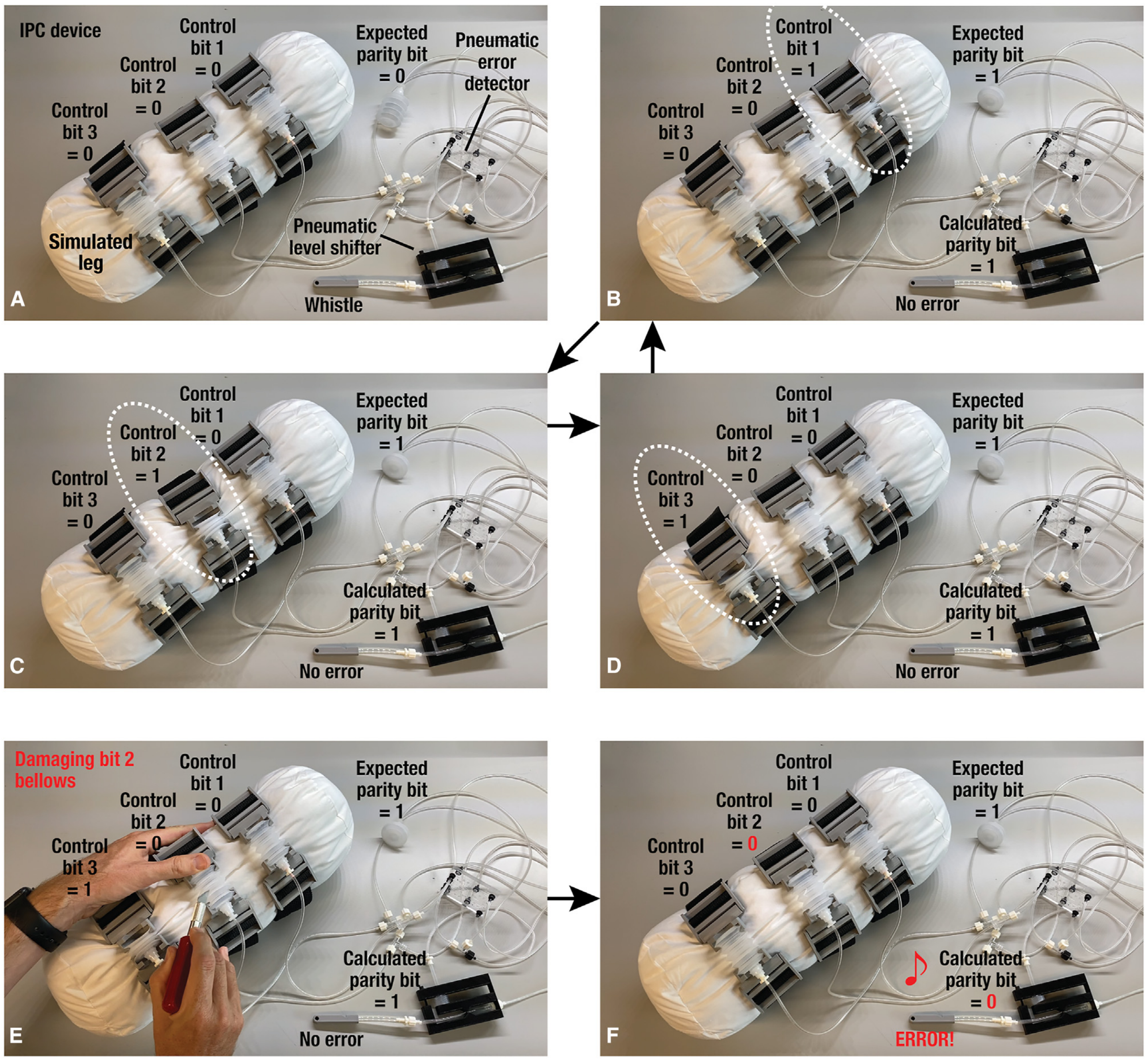


Figure 6. Using the pneumatic error detector to identify problems in a model medical device
Frames from Video S1 showing the pneumatic error detector monitoring the operation of a model soft-robotic medical device, an IPC device used to prevent blood clots in a wearer's legs (A). During normal operation (B → C → D → B → C → D …), the device control system contacts one bellows at a time and no errors are detected. However, when a bellows is punctured to create a leak (E), the pneumatic error detector recognizes the mismatch between the expected (**1**) and calculated (**0**) parity bit values and automatically alerts the wearer by blowing a whistle (F). Detailed explanations of each frame are in the main text.

pneumatic level shifter developed as part of another project. This level shifter (shown in Figure 6) consists of a small flexible plastic bellows mounted in a 3D-printed plastic frame. The bellows' motion is mechanically relayed to a pinch point through which runs flexible tubing connected to a pressurized air supply. When no error is detected by the attached error detector, the level shifter's bellows is at atmospheric pressure and is fully extended, holding the pinch point closed and blocking the flow of pressurized air in the tubing. However, when an error is detected, the level shifter's bellows receives vacuum from the error detector and contracts; this opens the pinch point and allows pressurized air to flow through the tubing and into the attached whistle, which makes a sound and alerts the wearer of a problem.

Demonstrating continuous error detection

We demonstrated error detection under two different modes of operation for the IPC device. In the first mode, the pneumatic error detector was operated after every change in the values of the control bits 1, 2, and 3. This mode offers continuous error checking (detecting an error as early as possible), but this comes at the expense of overall speed (the pneumatic error detector takes about 1 s to operate and about 5 s to reset after operation, so in this mode the control bits can only be updated every few seconds). A video recording of the IPC system in this mode of operation is available as Video S1; key snapshots are shown in Figure 6. Over several minutes of normal operation (repeated cycling through the states shown in Figures 6B, 6C, and 6D, operating the pneumatic error detector after each step), the error output remained at atmospheric pressure and the whistle remained silent; this is expected during normal error-free operation. Then, as shown in Figure 6E, we used a knife to puncture the IPC bellows connected to control bit 2. The next time that the system attempted to contract the damaged bellows by setting control bit 2 to **1** (vacuum) in Figure 6F, the vacuum was exhausted through the puncture in the bellows, so the error detector sensed that control bit 2 was **0** (atmospheric pressure). The error detector then used this value along with control bit 1 (**0**) and control bit 3 (**0**) to calculate a parity bit of **0** XOR **0** XOR **0** = **0**. This calculated value differed from the expected value of the parity bit input (**1**), which caused the error detector to output **1** (vacuum) to indicate the error. Finally, the pneumatic level shifter converted this signal to a positive pressure, which caused the whistle to blow (Figure 6F; see also Video S1). The whistle continued to sound every time that the error was detected again, until the leak was repaired. In this manner, the error detector successfully detected damage to the IPC device mere seconds after the damage occurred and notified the wearer about the problem.

Demonstrating periodic error detection

In the second mode of operation we demonstrated, the IPC device was alternated between two phases: a run phase, during which the control bits can be changed at high speeds without activating the pneumatic error detector, and a check phase, during which the pneumatic error detector checks each control bit in turn. This mode of operation offers periods of much faster operation (the control bits can be updated several times per second during the run phase) at the expense of error checking frequency (errors are only detected during the check phase). In Video S1, the IPC device alternated between spending 22.5 s in the run phase (during which a bellows was actuated every 750 ms) and 39 s in the check phase (during which the system applied vacuum to the control bits one at a time while the pneumatic error detector checked for errors). When the IPC device was damaged during the run phase by using scissors to cut the tubing leading to the bellows of control bit 3, the pneumatic error detector successfully sensed this damage and blew the whistle 45 s later during the system's next check phase. Finally, Video S1 also demonstrates that the pneumatic error detector's error signal is automatically reset after fixing the error. When we repaired the cut tubing, the whistle was again silent in subsequent check phases.

DISCUSSION

In this work we demonstrated that pneumatic logic can be used to detect failures in pneumatic systems. While this work focused primarily on a biomedical application for the pneumatic error detector, in principle, any pneumatically controlled system could gain sophisticated fault detection capabilities without sensors by adding a pneumatic logic circuit similar to the error detector. This simple and low-cost approach to error detection can promote safety and reliability across a wide range of important application areas. We conclude by discussing the advantages, limitations, and future directions for this technology.

Hardware requirements

The pneumatic error detector has favorable SWaP characteristics. The device is 6.35 cm wide and 6.25 mm thick, and its size could be reduced even further using smaller valves.[34] The error detector's 23 g represents a material cost of just $0.93 USD, and its design is amenable to automated manufacturing and mass production. Our current error detector does require some additional electromechanical control hardware to operate. Specifically, it needs one additional solenoid valve for providing the pneumatic error detector with the expected value of the parity bit and five additional solenoid valves for powering and resetting the pneumatic error detector. We discuss below possible ways to reduce or eliminate this electromechanical hardware. Even in its current form, the error detector provides a sophisticated level of error detection capabilities without adding electronic components to the pneumatic system being controlled. This is especially attractive for robotic applications in environments not suitable for electronics, such as in damp, explosive, and high-radiation areas.

Environmental and speed limitations

Air-powered logic circuits have disadvantages in some applications. For example, dust and soot particles can be drawn into the pneumatic logic circuit during operation. If these particles become trapped in a valve, then they can limit the ability of the valve to seal when closed and cause the device to malfunction. While porous self-adhesive tape can be applied to vent holes to serve as a filter and limit particle entry, pneumatic logic circuits may still not be suitable for use in particularly dusty or dirty environments. Additionally, pneumatic logic may be less suitable for use in high-speed systems. Operating and resetting the pneumatic error detector currently takes about 6 s, and while this could be optimized further, it nonetheless may not be feasible to use pneumatic logic to monitor rapidly changing pneumatic signals.

Pneumatic signal timing and synchronization

Pneumatic logic circuits are also susceptible to some of the same operational vulnerabilities that affect electronic logic circuits. For example, in electronic logic circuits, race conditions occur when variations in signal timing cause the circuit to behave differently and potentially give an incorrect answer. To illustrate this, consider an XOR gate in which both inputs are False; the output will also be False as expected for an XOR gate. If both inputs change to True at the same time, then the output will remain False. However, if one input changes a fraction of a second before the other input, then during that brief period of time between the first and second input changes, the XOR gate's inputs would be different (one False and one True), and the gate would output True. Thus, the behavior of an electronic XOR gate can be influenced by how much time it takes for input signals to reach the gate, a classic example of a race condition. In this work, we designed the pneumatic error detector to receive power vacuums and input signals all at the same time; if inputs reach the detector at slightly different times, device malfunctions can occur. We tried to reduce the risk of race condition-induced failures by keeping the channel lengths of the pneumatic error detector as consistent as possible. This is why the layouts of the detector's three XOR gates shown in blue in Figure 3E are identical or mirror images.

We also made the volumes of air contained in the pneumatic lines connected to the detector's inputs as similar as possible. These volumes are depressurized or repressurized with every change of the input signals, so if one input has a smaller (or larger) volume connected to it, its pressure will change faster (or slower) than the other inputs, and the resulting out-of-sync signals can cause a malfunction. This situation is analogous to an electronic circuit in which different capacitors take different amounts of time to charge or discharge under the same voltage. To avoid this situation, we included a bellows on the expected parity bit input in the IPC experiments shown in Figure 6. With it, all four inputs have one bellows attached and thus have similar volumes, so they all depressurize and repressurize at the same rate and ensure that their pneumatic signals reach the error detector at the same time. Finally, for some applications, it may not be feasible to keep the channel lengths and air volumes of the pneumatic input signals constant. In such cases, a pneumatic clock signal could be used to synchronize signals during device operation.[15]

Trapped vacuums

Another vulnerability that affects pneumatic logic circuits involves vacuum trapped inside the devices. In general, if P_1 or P_2 are under vacuum while P_C goes to atmospheric pressure, the valve will close and the vacuum in the valved channel will remain trapped in the channel unless it is vented by a source of atmospheric pressure somewhere else in the circuit (see Figures 2B and 2C). This is analogous to an electronic circuit containing capacitors that continue to store electric charges even after they are disconnected from a power supply. Trapped vacuums can be very useful in some situations; they can serve as pneumatic memory to control a larger number of microfluidic valves[12] or soft robotic actuators[23] using a smaller amount of hardware. However, in the pneumatic error detector, trapped vacuums are only a nuisance and they must be eliminated after every operation of the device to avoid malfunctions. The need to eliminate these trapped vacuums between operations adds considerable design and operational complexity to the pneumatic error detector. Two dedicated solenoid valves are required to momentarily apply vacuum to the reset inputs of the device to open valves S, T, and U and vent trapped vacuums, and this venting needs to occur after turning off the power vacuum and before turning off the input signals to avoid creating additional trapped vacuums. This reset sequence is described in greater detail in the experimental procedures below.

Eliminating the need to manually vent trapped vacuums could significantly reduce the amount of hardware and time required to operate the pneumatic error detector. In electronics, high-resistance bleeder resistors are used for this purpose. When a bleeder resistor is placed between a capacitor and ground, the small electric current that flows through the resistor does not interfere with normal device operation, but the resistor automatically discharges the capacitor when the device is disconnected from a power supply. This same approach could be used to automatically vent trapped vacuums in the pneumatic error detector by connecting vacuum-prone regions of the device to the atmosphere through long, thin, high-resistance channels. The small amount of airflow through these channels would not interfere with the normal operation of the error detector, but the channels would automatically vent vacuums that become trapped inside the device between operations. Using these bleeder channels in the pneumatic error detector could eliminate the need for vacuum-venting solenoid valves and significantly reduce the overall cost and complexity of the error detector system.

Limitations of parity-bit-based error detection

In its current form, the pneumatic error detector is limited in the types of errors it can detect. For example, if two control bits have wrong values at the same time, then the associated value of the parity bit would not change, and the pneumatic error detector would not be able to detect those simultaneous errors. This is a fundamental weakness of parity-bit-based error detectors, which cannot detect an even number of simultaneous errors. If multiple simultaneous errors are a realistic concern in an application, then pneumatic error detectors could be designed that use other error detection schemes. For example, algorithms such as cyclic redundancy checks[40] and Fletcher's checksum[41] can detect multiple simultaneous errors. While these algorithms are more complex than the parity bit approach shown here, the recent demonstration of a complete programmable computer using monolithic membrane valve-based pneumatic logic[15] shows that even complex computations can be performed in pneumatic logic circuits. Finally, the recent demonstration by Gopinathan et al.[42] of microfluidic transistors that operate on liquid instead of gas suggests that a hydraulic version of our error detector could be used to detect problems in hydraulic systems.

Limitations of vacuum-powered operation

The pneumatic error detector runs on vacuum. Using vacuum allows us to use monolithic membrane valves[26] as the transistors in our circuits; these vacuum-operated normally closed valves are generally far more amenable to use in complex pneumatic logic circuits[12,13,15,23,27–34] than pressure-operated normally open valves are. If the pneumatic system to be monitored also runs on vacuum, then the pneumatic error detector can be connected directly to the system being controlled. For systems that run on positive pressures, the simple 3D-printed pneumatic level shifter shown in Figure 6 could be used to convert signals between pressure and vacuum as necessary; research on this topic is ongoing.

Alternative error signals

In this proof of concept, we connected the output of the pneumatic error detector to a whistle to provide an audible notification of an error condition. The error output can be used for many other purposes, such as initiating a shutdown or restart of the

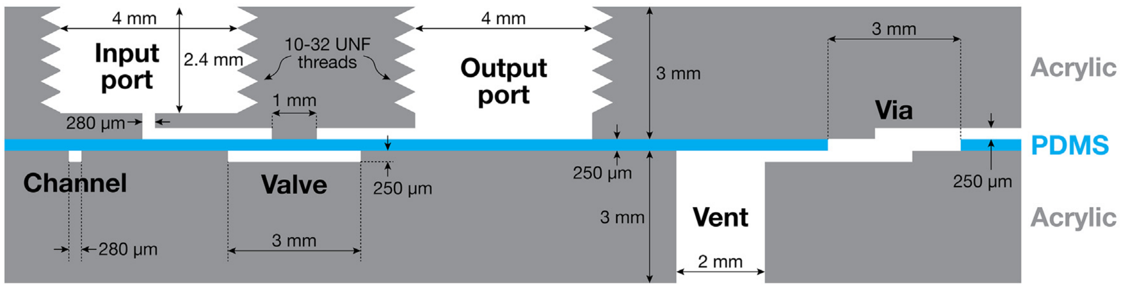


Figure 7. Cross-sectional scale diagram of key features of the pneumatic error detector

system being monitored. The outputs of multiple error detectors could be connected by OR gates to monitor a network of error detectors and provide a single unified error output. More sophisticated error detectors could provide the user with additional information about the detected error. For example, by connecting the error output to a pneumatic oscillator composed of an odd number of NOT gates arranged in a ring[27] and connecting one or more whistles to the outputs of the oscillator, a sequence of tones could be generated; the timing and pitches of these tones could provide information about the specific type of error that was detected.

EXPERIMENTAL PROCEDURES

Resource availability

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, William H. Grover (wgrover@engr.ucr.edu).

Materials availability

This study did not generate new unique reagents.

Data and code availability

All data generated in this study are included in the main text and the supplemental information. All original code is available in this paper's references and supplemental information. Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

Pneumatic error detector design and fabrication

The design of the pneumatic error detector was created in Adobe Illustrator (file available as Data S3) and exported as SVG files for milling into two acrylic substrates (each 6.35 cm wide × 5.08 cm high × 3 mm thick; Professional Plastics, Fullerton, CA) using a desktop CNC mill (Bantam Tools, Peekskill, NY). Channels were engraved at a width of 280 μm, and valve displacement chambers were engraved with a diameter of 3 mm; both features had a depth of 250 μm. Flat end mills (1/64 in. diameter, 2-flute; Bantam Tools) were used for engraving. Vents (locations where atmospheric-pressure air can enter the device) were milled as through holes with diameters of 2 mm, and the error output port was milled as a through hole with 4 mm diameter. The input ports (the control bits and expected parity bit inputs, the vacuum inputs, and the reset inputs) were milled with two diameters at different depths: 4 mm diameter for the first 2.4 mm, then 280 μm diameter for the remaining 0.6 mm. This narrowing at the bottom of the hole allows air to flow into the connected channel while preventing the PDMS membrane from being pulled into the input port by vacuum after the device is bonded. The input ports were then tapped with 10–32 unified fine pitch (UNF) threads (carbon steel plug tap 95007; Century Drill and Tool, Green Bay, WI).

The pneumatic error detector was bonded using a modified version of the protocol developed by Werner and colleagues.[43,44] First, the two pieces of acrylic were cleaned using 99.5% isopropanol and soaked in a solution of 3-aminopropyltriethoxysilane in purified water (5% by volume; Sigma-Aldrich, St. Louis, MO) for 20 min. The pieces were then rinsed in purified water and blown dry using nitrogen. Second, a 250-μm-thick sheet of PDMS (HT-6240; Bisco Silicones/Rogers Corporation, Carol Stream, IL) was cut out to the size of the acrylic pieces and punched with via holes (locations where the pneumatic signal needs to cross from one side of the PDMS membrane to the other) using a 3-mm-diameter biopsy punch (Electron Microscopy Sciences, Hatfield, PA). The bonding surfaces of both the acrylic and PDMS layers were then treated with a handheld corona treater (BD-20AC; Electro-Technic Products, Chicago, IL) for 1 min. Next, the acrylic and PDMS layers were assembled together to form the completed stack shown in Figure 2. The bonded device was clamped overnight using spring clamps (B01I0214J0, Amazon.com) to allow time for the bond to strengthen, then barbed tubing connectors (5463K2; McMaster-Carr, Santa Fe Springs, CA) were screwed into the ports, and excess PDMS was trimmed using a razor blade. A cross-sectional scale diagram showing the dimensions of key device features is shown in Figure 7.

IPC system design and fabrication

To test the pneumatic error detector with a model system representative of many soft robotic and medical applications, we designed and fabricated the model IPC device shown in Figure 6. Plastic bellows intended for dispensing applications (B08BZ1FRGH Kitchen Witch cake decorating set, Amazon.com) were connected via custom 3D-printed adapters (CAD files available as Data S4) to nylon webbing straps that encircle a simulated leg made from a fabric-covered cylinder of polyester batting. When vacuum is applied to one of the bellows, it contracts and squeezes the simulated leg.

Pneumatic level shifter design and fabrication

To convert the vacuum-based output of the pneumatic error detector to a positive pressure (suitable for blowing a whistle and other tasks), we designed and fabricated the pneumatic level shifter shown in Figure 6. The level shifter was fabricated using 3D printing (CAD files available as Data S5). A 16-mm-wide × 62-mm-long plastic bellows (B07HQ3N1HL, Amazon.com) was inserted in the conical recess in the level shifter, and laboratory tubing was seated in the pinch point in the level shifter. An M4-sized metric screw was threaded into the holes on the level shifter. This screw is used to adjust the compression between the bellows and the pinch point. With pressurized air applied to the laboratory tubing and atmospheric pressure applied to the bellows, the screw was tightened until the level shifter's pinch point stopped the flow of pressurized air through the tubing. The level shifter was then ready for use. Whenever vacuum is applied to the bellows, it contracts and opens the pinch point, which sends pressurized air through the tubing, but when atmospheric pressure is applied to the bellows, it expands and pinches the tubing closed again.

Pneumatic control and measurement system

A computer running LabVIEW (National Instruments, Austin, TX) and our OCW valve control software[39] was used to control a bank of nine two-way, three-ported solenoid valves (S070B-6BC, SMC Corporation of America; Noblesville, IN) via a digital control module (NI-9400 series, National Instruments).

Three solenoid valves provided vacuum (−68 kPa) or atmospheric pressure (0 kPa) to the model IPC device. The first solenoid valve was connected to the first bellows on the IPC device and the control bit 1 input on the pneumatic error detector. The second solenoid valve was connected to the second IPC bellows and control bit 2. The third solenoid valve was connected to the third IPC bellows and control bit 3. A fourth solenoid valve provided vacuum or atmospheric pressure to the expected parity bit input on the pneumatic error detector. Three solenoid valves provided vacuum to the "power" vacuum inputs on the pneumatic error detector (see Figure 3E). Finally, two solenoid valves provided vacuum to the "reset" inputs on the pneumatic error detector.

To characterize the performance of the pneumatic error detector, a custom Arduino-based multichannel pressure sensor circuit utilizing differential pressure gauges (MPX4250DP; NXP Semiconductors, Austin, TX) and a data-logging Python computer program were used to monitor the pressures at the three control bit inputs, one expected parity bit input, and one error bit output (data shown in Figures 5, S1, S2, and S3). The printed circuit board design and Arduino and Python code for the multichannel pressure monitor are available as Data S1 and S6.

Operating the pneumatic error detector

The pneumatic error detector is connected in parallel with the pneumatic system being controlled using tee junctions, as illustrated in Figure 1 and photographed in Figure 6. This ensures that the pneumatic signals in the system being controlled are also available to the control bit inputs of the error detector. Additionally, the pneumatic signal representing the current expected parity bit value is connected to the pneumatic error detector.

A typical experiment applies all 16 possible combinations of **1**s and **0**s (vacuum and atmospheric pressure) to the three control bit inputs and one expected parity bit input and monitors the pressure at the error bit output, as shown in Figures 5, S1, S2, and S3. For each combination of inputs, the system starts with all of the pneumatic error detector's inputs at atmospheric pressure (**0**s). Vacuum is then applied to the two power vacuum inputs and the subset of control and expected parity bit inputs that are to be set to **1**s in a given combination; the other inputs remain at atmospheric pressure (**0**). The power vacuum inputs and selected control and error bit inputs receive vacuum simultaneously. Once this occurs, the error detector automatically calculates the value of the parity bit corresponding to the current values of the control bits, compares the calculated parity bit to the expected parity bit, and applies a vacuum to the error bit output if the two parity bit values disagree. The calculation process takes about 1 s to complete. The error-indicating vacuum will continue for as long as the pneumatic error detector remains in this state.

To reset the pneumatic error detector, first, the power vacuum inputs are turned off in sequence (first power 3, then power 2, and finally power 1), with a brief 500-ms pause between them. Second, vacuum is momentarily applied to the detector's reset inputs: reset 1 receives vacuum for 500 ms and is then turned off, then reset 2 receives vacuum for 500 ms and is turned off. This briefly opens valves S, T, and U and vents regions in the device that may contain vacuums that would otherwise cause the device to malfunction during the next error detection cycle if not vented. Finally, the control bits and expected parity bit are reset to **0** (atmospheric pressure). This reset cycle takes about 5 s, after which the pneumatic error detector is again ready to detect errors. The OCW code used to perform the experiment shown in Figure 5 is provided in Data S2. For applications that are incompatible with the 5-s reset cycle (e.g., applications that require rapid actuation of the control bits), the system can alternate between run and check phases, as described above. During the run phase, the power vacuum inputs to the pneumatic error checker are kept off (at atmospheric pressure), so the control bits can be set to any desired pattern (and changed rapidly as desired) without activating the error detector. During the check phase, the vacuum supplies to the pneumatic error detector's "power" inputs are turned on and the control system sets the control and expected parity bits to whatever pattern is needed to check for errors (e.g., setting all control bits to **1** or vacuum, or setting each control bit to **1** in sequence). If an error is detected, then the pneumatic error detector will output a vacuum. Otherwise, once the check phase is completed and the error detector is reset, the system can reenter the run phase. In this manner, the system can alternate between run and check phases with whatever frequency is suitable for a given application.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.device.2024.100507.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation's Division of Civil, Mechanical, and Manufacturing Innovation under award nos. 1740052 (to P.B.) and 2046270 and 2133084 (to K.K.); the National Science Foundation's Division of Computing and Communication Frontiers under award no. 2019362 (to W.H.G.); and the National Science Foundation's Division of Biological Infrastructure under award no. 2131428 (to W.H.G.) (https://nsf.gov). The funders played no role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript. The authors thank an anonymous reviewer for suggesting that the duration and pitches of whistle tones could be used to provide additional information about the specific type of error detected by the pneumatic error detector.

AUTHOR CONTRIBUTIONS

S.H., M.S., Z.P., M.-H. T., and W.H.G. conceived the study, conducted the experiments, and analyzed the results. All authors contributed to the writing of the manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: March 11, 2024
Revised: March 12, 2024
Accepted: July 16, 2024
Published: August 12, 2024

REFERENCES

1. Zhang, X., Pan, T., Heung, H.L., Chiu, P.W.Y., and Li, Z. (2018). A biomimetic soft robot for inspecting pipeline with significant diameter variation. In RSJ International Conference on Intelligent Robots and Systems (IROS). Publisher: IEEE. ISBN 2153-0866, 7486–7491. https://doi.org/10.1109/IROS.2018.8594390.
2. Janghorban, A., and Dehghani, R. (2022). Design and motion analysis of a bio-inspired soft robotic finger based on multi-sectional soft reinforced actuator. J. Intell. Robot. Syst. *104*, 74. https://doi.org/10.1007/s10846-022-01579-3.
3. Zhao, S., Lei, Y., Wang, Z., Zhang, J., Liu, J., Zheng, P., Gong, Z., and Sun, Y. (2021). Biomimetic artificial joints based on multi-material pneumatic actuators developed for soft robotic finger application. Micromachines *12*, 1593. https://doi.org/10.3390/mi12121593.
4. Shao, Q., Dong, X., Lin, Z., Tang, C., Sun, H., Liu, X.J., and Zhao, H. (2022). Untethered robotic millipede driven by low-pressure microfluidic actuators for multi-terrain exploration. IEEE Robot. Autom. Lett. *7*, 12142–12149. https://doi.org/10.1109/LRA.2022.3213137.
5. Kokkoni, E., Liu, Z., and Karydis, K. (2020). Development of a Soft Robotic Wearable Device to Assist Infant Reaching. J. Eng. Sci. Med. Diagn. Ther. *3*, 021109. https://doi.org/10.1115/1.4046397.
6. Shi, L., Mucchiani, C., and Karydis, K. (2022). Online modeling and control of soft multi-fingered grippers via Koopman operator theory. In 2022 IEEE 18th International Conference on Automation Science and Engineering (CASE). Publisher: IEEE. 1946–1952. https://doi.org/10.1109/CASE49997.2022.9926464.
7. Liu, Z., Lu, Z., and Karydis, K. (2020). Sorx: A soft pneumatic hexapedal robot to traverse rough, steep, and unstable terrain. In 2020 IEEE International Conference on Robotics and Automation (ICRA). Publisher: IEEE. 420–426. https://doi.org/10.1109/ICRA40945.2020.9196731.

8. Seyidoğlu, B., and Rafsanjani, A. (2024). A textile origami snake robot for rectilinear locomotion. Device *2*, 100226. https://doi.org/10.1016/j.device.2023.100226.
9. (1895). Heat-regulating apparatus. US Patent No. 542733.
10. Ord-Hume, A.W.J.G. (1985). Pianola: The History of the Self-Playing Piano (George Allen & Unwin).
11. Gluskin, R.S., Jacoby, M., and Reader, T.D. (1964). FLODAC: a pure fluid digital computer. In *Proceedings of the October 27-29, 1964, Fall Joint Computer Conference, Part I*, AFIPS '64 (Fall, part I) (Association for Computing Machinery), pp. 631–641. https://doi.org/10.1145/1464052.1464112.
12. Grover, W.H., Ivester, R.H.C., Jensen, E.C., and Mathies, R.A. (2006). Development and multiplexed control of latching pneumatic valves using microfluidic logical structures. Lab Chip *6*, 623–631. https://doi.org/10.1039/b518362f.
13. Jensen, E.C., Grover, W.H., and Mathies, R.A. (2007). Micropneumatic digital logic structures for integrated microdevice computation and control. J. Microelectromech. Syst. *16*, 1378–1385. https://doi.org/10.1109/JMEMS.2007.906080.
14. Rhee, M., and Burns, M.A. (2009). Microfluidic pneumatic logic circuits and digital pneumatic microprocessors for integrated microfluidic systems. Lab Chip *9*, 3131–3143. https://doi.org/10.1039/B904354C.
15. Ahrar, S., Raje, M., Lee, I.C., and Hui, E.E. (2023). Pneumatic computers for embedded control of microfluidics. Sci. Adv. *9*, eadg0201. https://doi.org/10.1126/sciadv.adg0201.
16. Singh, A.P., Tintelott, M., Moussavi, E., Ingebrandt, S., Leupers, R., Vu, X.T., Merchant, F., and Pachauri, V. (2023). Logic operations in fluidics as foundation for embedded biohybrid computation. Device *1*, 100220. https://doi.org/10.1016/j.device.2023.100220.
17. Rothemund, P., Ainla, A., Belding, L., Preston, D.J., Kurihara, S., Suo, Z., and Whitesides, G.M. (2018). A soft, bistable valve for autonomous control of soft actuators. Sci. Robot. *3*, eaar7986. https://doi.org/10.1126/scirobotics.aar7986.
18. Drotman, D., Jadhav, S., Sharp, D., Chan, C., and Tolley, M.T. (2021). Electronics-free pneumatic circuits for controlling soft-legged robots. Sci. Robot. *6*, eaay2627. https://doi.org/10.1126/scirobotics.aay2627.
19. Lee, W.K., Preston, D.J., Nemitz, M.P., Nagarkar, A., MacKeith, A.K., Gorissen, B., Vasios, N., Sanchez, V., Bertoldi, K., Mahadevan, L., and Whitesides, G.M. (2022). A buckling-sheet ring oscillator for electronics-free, multimodal locomotion. Sci. Robot. *7*, eabg5812. https://doi.org/10.1126/scirobotics.abg5812.
20. Zhai, Y., De Boer, A., Yan, J., Shih, B., Faber, M., Speros, J., Gupta, R., and Tolley, M.T. (2023). Desktop fabrication of monolithic soft robotic devices with embedded fluidic control circuits. Sci. Robot. *8*, eadg3792. https://doi.org/10.1126/scirobotics.adg3792.
21. Preston, D.J., Rothemund, P., Jiang, H.J., Nemitz, M.P., Rawson, J., Suo, Z., and Whitesides, G.M. (2019). Digital logic for soft devices. Proc. Natl. Acad. Sci. USA *116*, 7750–7759. https://doi.org/10.1073/pnas.1820672116.
22. Preston, D.J., Jiang, H.J., Sanchez, V., Rothemund, P., Rawson, J., Nemitz, M.P., Lee, W.K., Suo, Z., Walsh, C.J., and Whitesides, G.M. (2019). A soft ring oscillator. Sci. Robot. *4*, eaaw5496. https://doi.org/10.1126/scirobotics.aaw5496.
23. Hoang, S., Karydis, K., Brisk, P., and Grover, W.H. (2021). A pneumatic random-access memory for controlling soft robots. PLoS One *16*, e0254524. https://doi.org/10.1371/journal.pone.0254524.
24. Hoang, S., Shehada, M., Karydis, K., Brisk, P., and Grover, W.H. (2024). Controlling biomedical devices using pneumatic logic. medRxiv. https://doi.org/10.1101/2024.01.24.24301744.
25. Jumet, B., Zook, Z.A., Yousaf, A., Rajappan, A., Xu, D., Yap, T.F., Fino, N., Liu, Z., O'Malley, M.K., and Preston, D.J. (2023). Fluidically programmed wearable haptic textiles. Device *1*, 100059. https://doi.org/10.1016/j.device.2023.100059.
26. Grover, W.H., Skelley, A.M., Liu, C.N., Lagally, E.T., and Mathies, R.A. (2003). Monolithic membrane valves and diaphragm pumps for practical large-scale integration into glass microfluidic devices. Sensor. Actuator. B Chem. *89*, 315–323. https://doi.org/10.1016/S0925-4005(02)00468-9.
27. Duncan, P.N., Nguyen, T.V., and Hui, E.E. (2013). Pneumatic oscillator circuits for timing and control of integrated microfluidics. Proc. Natl. Acad. Sci. USA *110*, 18104–18109. https://doi.org/10.1073/pnas.1310254110.
28. Jensen, E.C., Bhat, B.P., and Mathies, R.A. (2010). A digital microfluidic platform for the automation of quantitative biomolecular assays. Lab Chip *10*, 685–691. https://doi.org/10.1039/b920124f.
29. Jensen, E.C., Zeng, Y., Kim, J., and Mathies, R.A. (2010). Microvalve enabled digital microfluidic systems for high performance biochemical and genetic analysis. JALA Charlottesv. Va. *15*, 455–463. https://doi.org/10.1016/j.jala.2010.08.003.
30. Jensen, E.C., Stockton, A.M., Chiesl, T.N., Kim, J., Bera, A., and Mathies, R.A. (2013). Digitally programmable microfluidic automaton for multiscale combinatorial mixing and sample processing. Lab Chip *13*, 288–296. https://doi.org/10.1039/c2lc40861a.
31. Kim, J., Jensen, E.C., Stockton, A.M., and Mathies, R.A. (2013). Universal microfluidic automaton for autonomous sample processing: application to the Mars Organic Analyzer. Anal. Chem. *85*, 7682–7688. https://doi.org/10.1021/ac303767m.
32. Linshiz, G., Jensen, E., Stawski, N., Bi, C., Elsbree, N., Jiao, H., Kim, J., Mathies, R., Keasling, J.D., and Hillson, N.J. (2016). End- to-end automated microfluidic platform for synthetic biology: from design to functional analysis. J. Biol. Eng. *10*, 3. https://doi.org/10.1186/s13036-016-0024-5.
33. Nguyen, T.V., Duncan, P.N., Ahrar, S., and Hui, E.E. (2012). Semi- autonomous liquid handling via on-chip pneumatic digital logic. Lab Chip *12*, 3991–3994. https://doi.org/10.1039/c2lc40466d.
34. Duncan, P.N., Ahrar, S., and Hui, E.E. (2015). Scaling of pneumatic digital logic circuits. Lab Chip *15*, 1360–1365. https://doi.org/10.1039/c4lc01048e.
35. Lukolf, H. and Welsh, H.F. (1952). The UNISERVO-tape reader and recorder. In International Workshop on Managing Requirements Knowledge. IEEE Computer Society, 47. https://doi.org/10.1109/AFIPS.1952.28.
36. Partsch, H. (2008). Intermittent pneumatic compression in immobile patients. Int. Wound J. *5*, 389–397.
37. Morris, R.J. (2008). Intermittent pneumatic compression - systems and applications. J. Med. Eng. Technol. *32*, 179–188. https://doi.org/10.1080/03091900601015147.
38. Chen, A.H., Frangos, S.G., Kilaru, S., and Sumpio, B.E. (2001). Intermittent pneumatic compression devises - physiological mechanisms of action. Eur. J. Vasc. Endovasc. Surg. *21*, 383–392. https://doi.org/10.1053/ejvs.2001.1348.
39. Grover, W.H. (2024). OCW: A really simple language for controlling microfluidic valves. https://github.com/groverlab/ocw.
40. Peterson, W., and Brown, D. (1961). Cyclic codes for error detection. Proc. IRE *49*, 228–235. https://doi.org/10.1109/JRPROC.1961.287814.
41. Fletcher, J. (1982). An arithmetic checksum for serial transmissions. IEEE Trans. Commun. *30*, 247–252.
42. Gopinathan, K.A., Mishra, A., Mutlu, B.R., Edd, J.F., and Toner, M. (2023). A microfluidic transistor for automatic control of liquids. Nature *622*, 735–741. https://doi.org/10.1038/s41586-023-06517-3.
43. Werner, E.M. and Hui, E.E. (2022). Microfluidic digital logic chip assembly. protocols.io. 10.17504/protocols.io.8efhtbn.
44. Werner, E.M., Lam, B.X., and Hui, E.E. (2022). Phase-optimized peristaltic pumping by integrated microfluidic logic. Micromachines *13*, 1784. https://doi.org/10.3390/mi13101784.