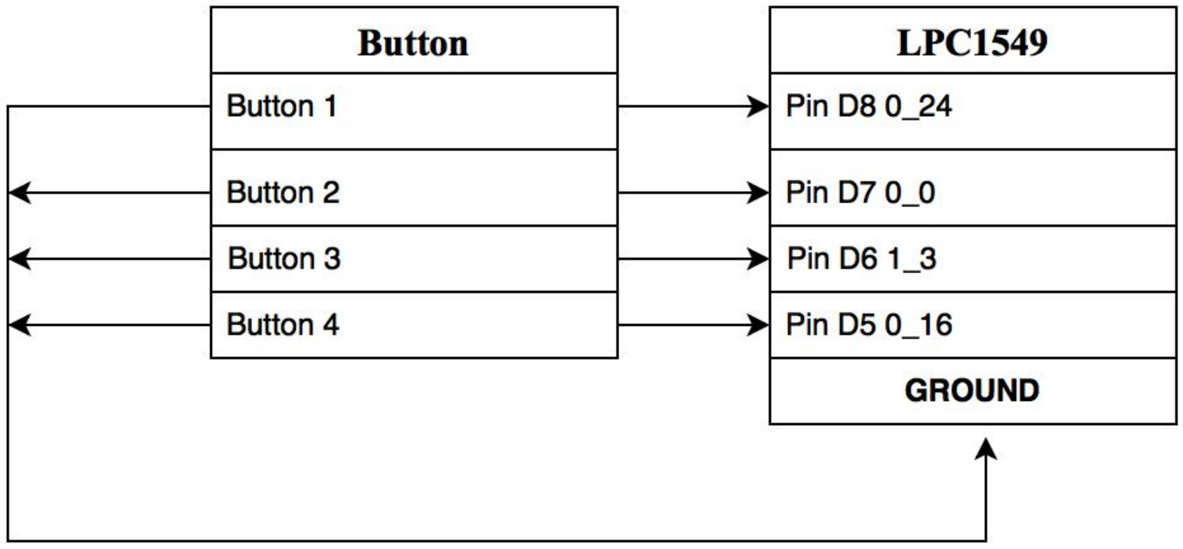
Introduction

Team member: Trung Ngo, Nhan Phan.

In this project, our goal is to control the ventilation fan to reach the target pressure using PID control. We also added voice control to the project.

Button wiring diagram



User manual

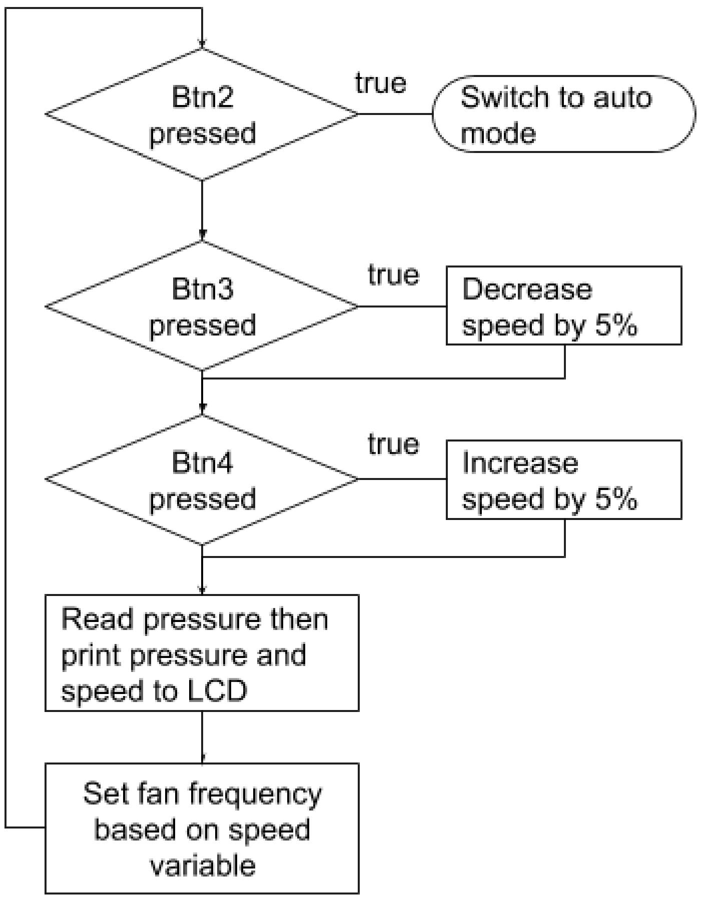
Button (from left to right)	Automatic mode	Manual mode
Button 1	Change step from 1-2-3	Change step from 5%-10%-15%
Button 2	Change to Manual mode	Change to Automatic mode
Button 3	Lower target pressure by 1 step	Lower fan speed by 1 step
Button 4	Raise target pressure by 1 step	Raise fan speed by 1 step

For voice command, when the user speaks the key word, LPC1549 wil do the following command (the system have a delay around 1~2 second for processing):

Voice keyword	Automatic mode	Manual mode
Off | Shut	Turn system off	
On | Start	Turn system on	
Step + 1 to 10	Change step from 1 to 10	
20 to 100 (divisible by 10)	Set target pressure to said number	Set current speed to said number

Manual mode

The manual mode controller is implemented as a simple poll loop waiting on button input then set the speed or state accordingly to the button pressed.



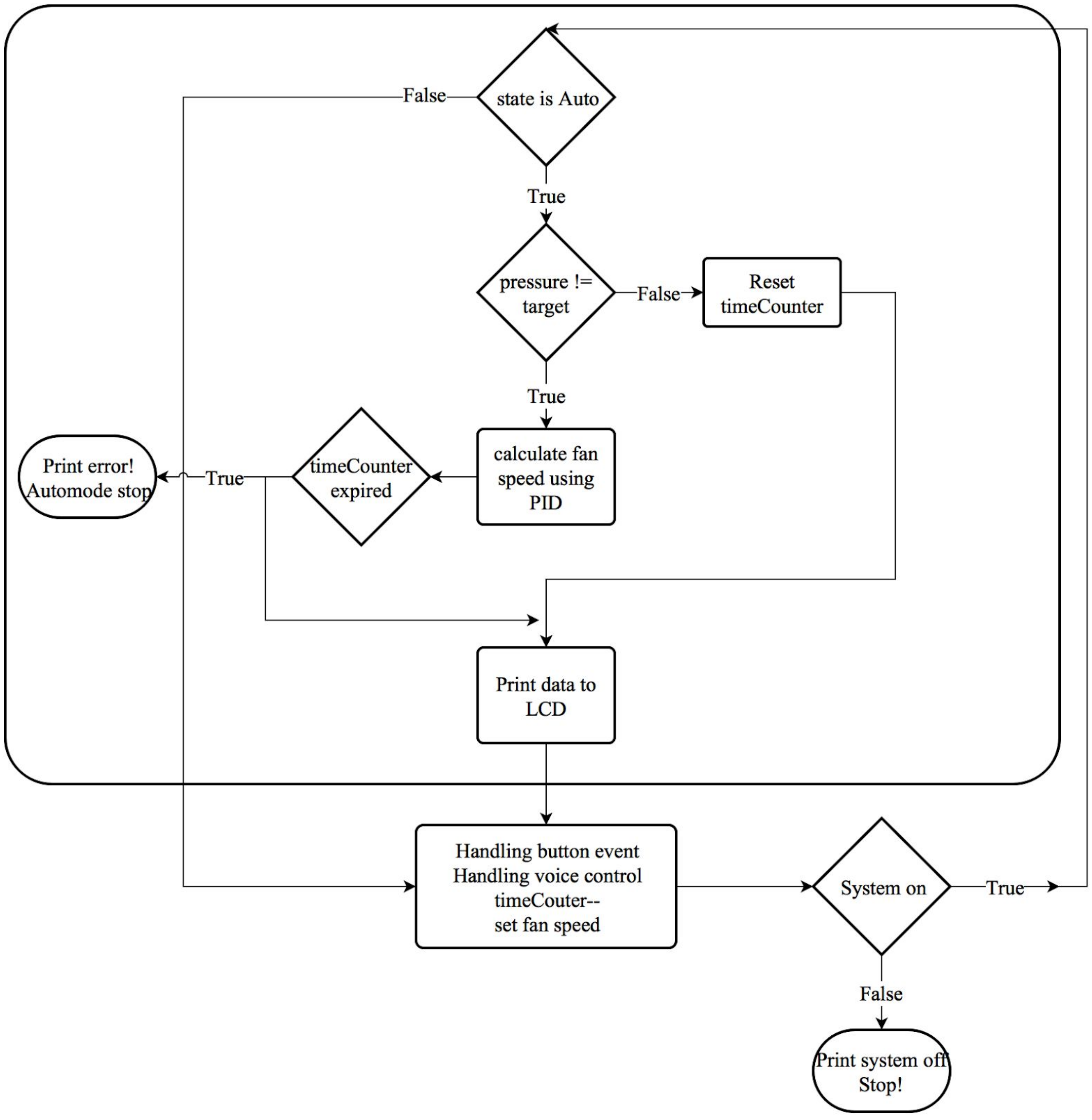
Automatic mode

Control using PID. With Kp = 120, Ki = 10, Kd = 100. Using the formula:

```
Speed = Kp*error + Ki*integral + Kd*derivative
```

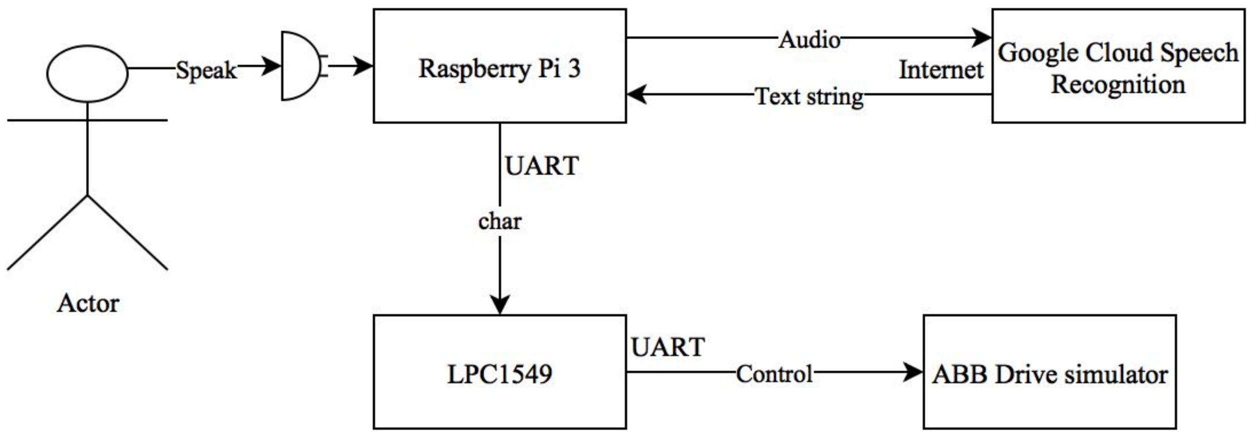
Fan speed value is clamped to the range of 0-20,000.
The total delay for 1 loop is around 2 milliseconds. The current pressure is measured using median value of the 11 recent pressure result from sensor.

Code diagram



Voice control

Concept of voice control



First, I need to set up Raspberry Pi 3 so it can use Google Cloud API. After that, using Python, I make a while loop in which Raspberry Pi keep trying to record sound from the microphone and send to Google Cloud to get the response from their Speech Recognition service.

Google Cloud Speech Recognition service is a premium service (you have to pay for it), so to avoid spending over the free quota, I only record the sound for 3 seconds each time.

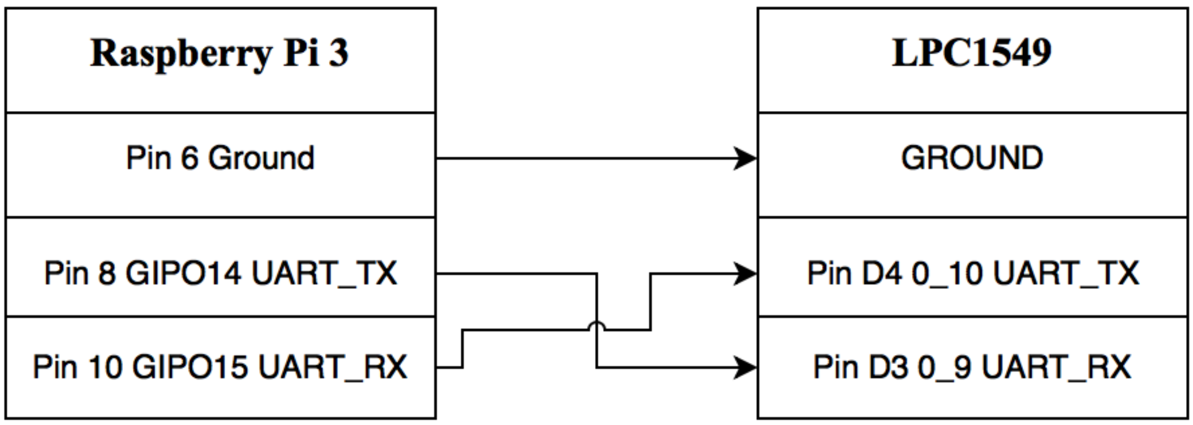
To avoid sending meaningless file to server, I set a minimum loudness level before the audio can be processed

I can also add trigger words ("Alexa" or "Echo" or my own choice), however, trigger words mean bigger data.

If Raspberry Pi is connect with a loudspeaker, it can also play sound such as "I'm listening…" or "Sorry, I don't understand".

To send data from Raspberry Pi 3 to LPC1549, I used UART. LPC1549 has 3 UARTs, however, the project already uses UART1 to connect to the Arduino controller. So I modified the "SerialPort" class to use UART2 to send data.

UART Wiring diagram



Result

In automatic mode, it's take around 6~7 seconds to reach the target from start.

The pressure sensor have some problem. When you close the cap, and start to lower fan speed from 30%. When to lower from 15% to 10%, pressure should be lower. Instead, the pressure will rise to 24~25 Pa first before reaching the stable value of 4 Pa. The pressure sensor behaves differently if the fan speed is lower than 10%, which make it impossible to set the target pressure at level lower than 10 Pa.

The set fan speed also have problem if the target pressure set too low (<10 Pa). The fan speed go up and down quickly until it now working anymore. Then the fan speed keep running at about 40%, and any further attempt to change the speed to 0 will not work. If the fan speed is stuck, then manual raise the fan speed up to 15% in manual mode will reset it. So far we cannot find a solution for this problem.