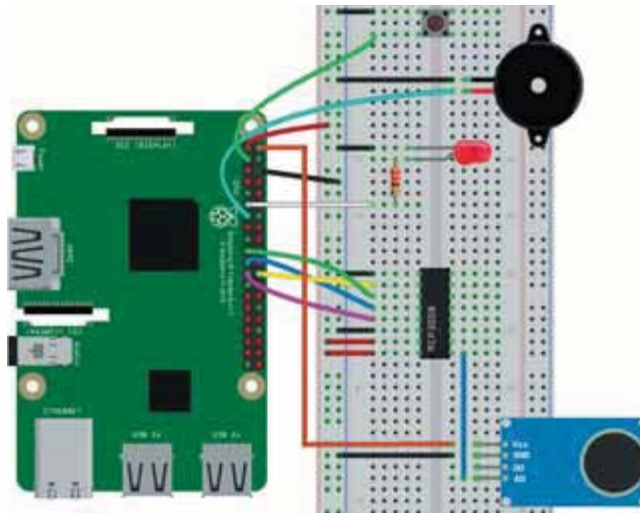
Rysunek 11.3
Diagram obwodu detektora dymu i gazu



1. Łączymy GND z niebieską szyną płytki montażowej, a 3,3 V z czerwoną szyną.
2. Umieszczamy chip MCP3008 na środku płytki montażowej, tak aby jego nóżki były po obu stronach środkowego podziału, jak pokazano na rysunku 11.3 i łączymy go przewodami według poniższej tabeli.

MCP3008	POŁĄCZYĆ Z
1	MQ-2 Pin AO
9	GND
10	GPIO 8
11	GPIO 10
12	GPIO 9
13	GPIO 11
14	GND
15	3,3 V
16	3,3 V

Pamiętajmy, że gdy półkole na MCP3008 jest na górze, pin 1 jest górnym pinem po lewej stronie; pełny układ pinów MCP3008 można znaleźć w punkcie „Przetworniki analogowo-cyfrowe” na stronie 57.

3. Umieszczamy czujnik gazu i dymu MQ-2 na płytce montażowej i łączymy przewodami według wskazówek.

CZUJNIK MQ-2	POŁĄCZYĆ Z
VCC	5 V
GND	GND
DO	Bez połączenia
AO	Pin 1 MCP3008

4. Wstawiamy LED w płytkę montażową. Łączymy dodatnie wyprowadzenie z pinu 17 GPIO przez rezystor 330 Ω i łączymy ujemne wyprowadzenie z szyną GND.
5. Wstawiamy przycisk na środku płytki montażowej z dwoma wyprowadzeniami po obu stronach centralnego podziału. Łączymy dolne prawe wyprowadzenie z szyną zasilania GND i dolne lewe wyprowadzenie z GPIO 2, upewniając się, że oba połączone wyprowadzenia są po tej samej stronie podziału.
6. Wstawiamy brzęczyk w płytkę montażową i łączymy czarny przewód z GND, a czerwony z GPIO 27.

Po połączeniu obwodu przewodami czas na przesłanie kodu.

PISANIE SKRYPTU

Otwieramy **Python 3 (IDLE)** i przechodzimy do **File ▸ New File**, aby utworzyć nowy skrypt. Kopiujemy kod z listingu 11.1 do edytora Pythona i zapisujemy skrypt jako *smoke_detector.py w folderze Czujniki.* (Pamiętajmy, że wszystkie skrypty można pobrać na stronie *https://smartkids.pwn.pl/ksiazka/20-prostych-projektow-raspberry-pi/*):

Listing 11.1
Skrypt detektora dymu i gazu

```
#import potrzebnych bibliotek
❶ from gpiozero import LED, Button, Buzzer, MCP3008
from time import sleep

❷ led = LED(17)
button = Button(2)
buzzer = Buzzer(27)
czujnik_gazu = MCP3008(0)

❸ stan_czujnika_gazu = False

❹ prog = 0.1

❺ def uzbroj_czujnik_gazu():
    global stan_czujnika_gazu
    if stan_czujnika_gazu == True:
        stan_czujnika_gazu = False
```

```
            led.off()
        else:
            stan_czujnika_gazu = True
            led.on()
❻ button.when_pressed = uzbroj_czujnik_gazu

❼ while True:
❽     #print(czujnik_gazu.value)
      #sprawdzanie, czy czujnik gazu jest uzbrojony
      #i czy osiągnął wartość progową
      if(stan_czujnika_gazu == True and czujnik_gazu.value >
prog):
          buzzer.beep()
      else:
          buzzer.off()
      sleep(2)
```

Najpierw importujemy klasy `LED`, `Button`, `Buzzer` and `MCP3008` z biblioteki gpiozero i funkcję `sleep` z biblioteki time ❶; następnie tworzymy obiekty `gpiozero`, które odnoszą się do przycisku LED, MCP3008 (czujnik gazu MQ-2) i brzęczyka ❷. Następnie tworzymy zmienną `stan_czujnika_gazu`, która wskazuje, czy czujnik dymu jest uzbrojony ❸; czujnik jest uzbrojony, gdy ta zmienna jest `True`, a nieuzbrojony, gdy `False`. Trzeba ustawić wartość `prog`, tak aby brzęczyk wydawał dźwięki tylko wtedy, gdy poziomy gazów są powyżej tego progu ❹. Za chwilę zajmiemy się określaniem wartości tego progu.

Funkcja `uzbroj_czujnik_gazu()` ❺ uzbraja i rozbraja czujnik, przełączając wartość zmiennej `stan_czujnika_gazu` na przeciwną, niezależnie od tego, czy przy wywołaniu funkcji aktualną wartością jest `True` czy `False`. W wierszu ❻ ustawiamy funkcję, która ma być wywoływana, gdy przycisk zostanie naciśnięty, aby można było ręcznie uzbroić i rozbroić czujnik. Możemy także tak ustawić diodę LED, aby ją włączać, gdy czujnik będzie uzbrojony. W ten sposób można wizualnie rozpoznać jego stan.

Ostatnim blokiem kodu jest pętla `while` ❼, która wciąż sprawdza, czy czujnik jest uzbrojony i czy poziomy gazów są powyżej progu. Jeśli czujnik jest uzbrojony i poziomy gazów są powyżej progu, brzęczyk wydaje dźwięki za pośrednictwem funkcji `buzzer.beep()`. Ostatnia funkcja `buzzer.off()` zatrzymuje brzęczyk.

Ustawianie wartości progu

Aby dokładnie ustawić bezpieczny poziom gazu, najpierw trzeba skalibrować czujnik z otoczeniem. To oznacza, że musimy pomierzyć