Złożenie pracy online:
2018-03-16 08:53:21
Kod pracy:
8451/36790/CloudA

Jakub Łatka
(nr albumu: 4116)

Praca inżynierska

Wykorzystanie technologii inteligentnych instalacji - sterownik akwarystyczny

The use of smart installations – an aquarium control device

Wydział: Wydział Nauk Społecznych i Informatyki

Kierunek: Informatyka

Specjalność: grafika 3d i multimedia

Promotor: dr Henryk Telega

Niniejszą pracę pragnę zadedykować całej mojej rodzinie, w szczególności żonie Annie i wspaniałym dzieciom Maciusiowi i Justynce, dzięki którym miałem możliwość kształcić się i zdobywać cenną wiedzę, którzy stale mnie wspierali i motywowali przez okres trwania studiów.

Za cierpliwość i wyrozumiałość przy realizacji niniejszej pracy pragnę złożyć podziękowania mojemu promotorowi dr Henrykowi Teledze.

Streszczenie

W niniejszej pracy opisano w jaki sposób autor wykorzystując zasady i technologie inteligentnych instalacji budynkowych samodzielnie zaprojektował i wykonał inteligentny system (sterownik), dzięki któremu będzie zarządzał sztucznym zbiornikiem wodnym – akwarium.
Przy realizacji sterownika wykorzystano platformę Raspberry Pi, oraz Arduino UNO, w celu kontrolowania i zarządzania całym zbiornikiem. System zakłada między innymi automatyczną obsługę oświetlenia, temperatury i poziomu PH wody. Dodatkowo, odpowiednie oprogramowanie sterownika pozwala utrzymywać optymalne warunki fauny i flory, a co więcej, zapewnia estetyczne wrażenia właścicielowi. Wreszcie, dzięki zastosowaniu sterownika zrealizowane zostanie automatyczne karmienie ryb, nawożenie, poprzez sterowanie pompami dozującymi oraz będzie on pełnił funkcję ochronną (ze względu na zastosowany system ostrzegania przed zalaniem).
Działanie sterownika zaimplementowano w języku Python, a także w pseudo C (w przypadku Arduino). Sterownik obsługiwany jest z poziomu przeglądarki www, natomiast sam interface zaimplementowany jest w PHP. Wszelkie dane potrzebne do działania przechowywane są w bazie danych (MariaDB). Ponadto, do efektywnego działania sterownika, autor wykorzystał różnego rodzaju elementy układów elektronicznych takie jak oporniki i potencjometry, a także przekaźniki.
Inteligentne instalacje można z powodzeniem zaadaptować zarówno w budynkach, jak i mniejszych obiektach. W przypadku akwarium - zastosowanie w sterowniku różnego rodzaju czujników oraz określonego sposobu programowania, pozwoliło na automatyzację zarządzania, co przyczyniło się do oszczędności czasu oraz kosztów.

Słowa kluczowe

inteligentne instalacje, Raspberry Pi, Arduino, sterownik, Python, Php, baza danych, automatyka

Abstract

This dissertation describes the process of designing and building a smart control system for an artificial water tank – aquarium, with the use of the rules and technologies of smart building installations.
Raspberry Pi and Arduino UNO platforms were used to create the control device, which enabled the control and management of the entire tank. Among other functionalities, the system automatically adjusts the lighting, temperature, and pH level of water. Furthermore, appropriate driver software makes it possible to maintain optimal conditions for the fauna and flora, providing appealing visual experiences for the owner. Finally, with the use of the control device fish are automatically fed, water is fertilized, and by controlling the operation of dosing pumps it also has a protective function (thanks to the overflow warning system applied).
The control device was implemented based on Python programming language, as well as pseudo-C (for Arduino). The driver is controlled via a web browser, and the interface was implemented based on PHP. All data required for the operation is stored in the database (MariaDB). What is more, to ensure efficient work of the driver, the author used various components of electronic circuits, such as resistors, potentiometers, and relays.
Smart installations can be successively applied both in buildings, and in smaller objects. In the case of an aquarium, using different sensors along with a particular programming method allowed for management automation, which resulted in noticeable savings in time and costs.

Keywords

inteligent installations, Raspberry Pi, Arduino, control device, Python, Php, database, automation

1 Wstęp, cel pracy

Celem niniejszej pracy jest zaprojektowanie i realizacja inteligentnego systemu (sterownika), który będzie zarządzał sztucznym zbiornikiem wodnym - akwarium. System będzie sterował m.in. oświetleniem, temperaturą i poziomem PH wody. Dodatkowo, odpowiednie oprogramowanie sterownika będzie utrzymywać optymalne warunki fauny i flory, jak również zrealizowane zostanie automatyczne karmienie ryb i nawożenie poprzez sterowanie pompami dozującymi oraz będzie on pełnił funkcję ochronną (ze względu na zastosowany system ostrzegania przed zalaniem). Przy realizacji zostaną wykorzystane zasady i technologia związana z inteligentnym zarządzaniem budynkami.

Technologie zintegrowanych inteligentnych systemów zarządzania[1] pozwalają na sterowanie pracą większości urządzeń elektrycznych w budynkach mieszkalnych i innych, ale z powodzeniem można je wykorzystać również w mniejszych obiektach.

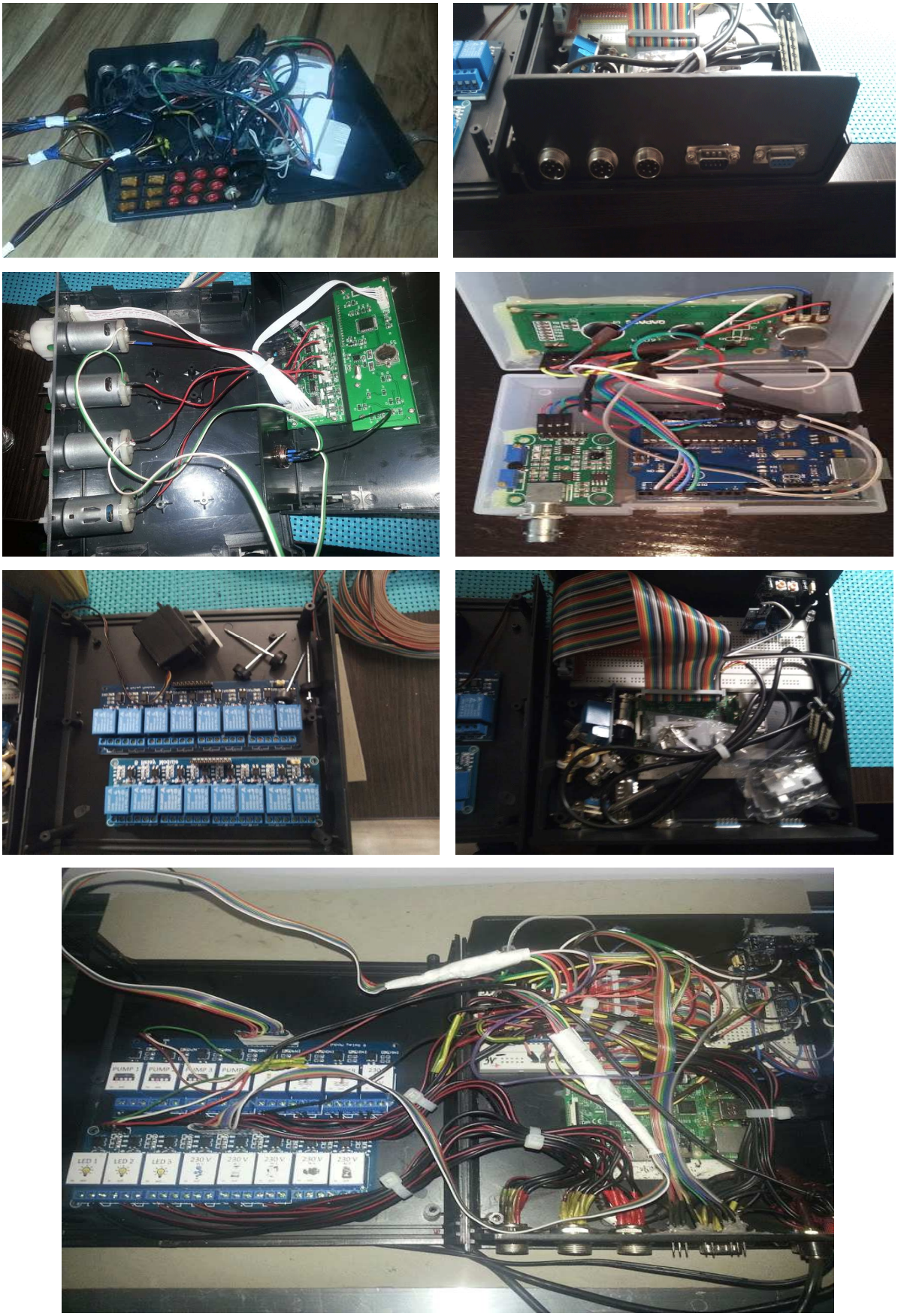
Koszty takich inteligentnych instalacji stanowią największą barierę, która skutecznie ogranicza ich codzienne stosowanie przez użytkowników domowych, a w dobie obecnej zaawansowanej technologii możliwość wykorzystania takich narzędzi powinna być standardem.

Sterownik akwarystyczny zagwarantuje, że panujące w zbiorniku warunki będą utrzymywane na stałym, odpowiednim poziomie, korzystnym zarówno dla zwierząt, jak i roślin. Dodatkowo sterownik zostanie wykorzystany do zarządzania osprzętem i ozdobami wchodzącymi w skład akwarium.

Sercem sterownika będzie platforma Raspberry Pi wspomagana przez Arduino. W ramach niniejszej pracy autor w całości samodzielnie zaprojektował i zbudował sterownik. Poniżej znajdują się zdjęcia dokumentujące budowę urządzenia.

[1] Dechnik M., Furtak M.: Inteligentne Budynki Dziś i Jutro Czasopismo "Builder", czerwiec 2017 str. 64-67

Rys. 1.1. zdjęcia podczas budowy urządzenia



Źródło: fotografie własne

2 Założenia systemu

Sterownik akwarystyczny będzie zarządzał całym zbiornikiem. System zakłada m.in. automatyczną obsługę oświetlenia poprzez wykorzystanie czujnika światła (patrz rozdział: 5.2). Utrzymanie zadanej temperatury będzie możliwe dzięki wykorzystaniu termostatu (patrz rozdział: 5.3), sterującego pracą przewodu grzewczego (patrz Rys. 2.1) rozłożonego na całej powierzchni zbiornika oraz lodówki przepływowej (patrz Rys. 2.2).

Rys. 2.1. przewód grzewczy (zdjęcie poglądowe)

Źródło: http://www.aquastyl.pl/c3c1da28b8b43b06f40fcb3e9f50cf19,Przewód_grzewczy_80_W,opis.html (data odczytu: 01-12-2017 r.)

System ma zadbać również o bezpieczeństwo poprzez zastosowanie systemu ostrzegania przed zalaniem (patrz rozdział: 5.5). Sterownik będzie samodzielnie utrzymywał poziom PH wody w zbiorniku poprzez jego pomiar (patrz rozdział: 5.4) oraz sterowanie

Rys. 2.2. chłodziarka akwariowa (zdjęcie poglądowe)

Źródło: https://www.zooantus.pl/home/120-chlodziarka-resun-cl200.html (data odczytu: 01-12-2017 r.)

Rys. 2.3. butla z CO_2 wyposażona w reduktor i elektrozawór

Źródło: zdjęcie własne

Rys. 2.4. piaskospad (zdjęcie poglądowe)

Źródło: https://sklep.wodna-kraina.pl/tylko-u-nas-piaskospad-do-akwarium-15x15-5x25cm-nowosc-wodna-kraina (data odczytu: 12-12-2017 r.)

elektrozaworem systemu nawożenia CO_2 (patrz Rys. 2.3) i napowietrzaczem, który także będzie można regulować za pomocą serwomechanizmu (patrz rozdział: 5.9). Zrealizowane zostanie także automatyczne karmienie mieszkańców akwarium (patrz rozdział: 5.8), nawożenie poprzez sterowanie pompami (patrz rozdział: 5.7) oraz obsługa uruchamianiem ozdób (patrz Rys. 2.4). Całość będzie obsługiwana z poziomu przeglądarki www za pomocą interface'u zaimplementowanego w PHP w połączeniu z bazą MariaDB (patrz rozdział: 6).

3 Inteligentne instalacje

W instalacjach budynków inteligentnych wykorzystywane są różnego rodzaju czujniki, które dostarczają szeregu informacji, dzięki którym można zarządzać wieloma urządzeniami. W praktyce można sterować między innymi:

- temperaturą (ogrzewaniem, klimatyzacją),
- oświetleniem,
- urządzeniami codziennego użytku,
- systemami alarmowymi, monitoringu, przeciwpożarowymi.

Zastosowanie inteligentnych instalacji wpływa na oszczędność czasu i pieniędzy, bezpieczeństwo mienia i użytkowników, a także podniesienie komfortu życia.

1. Oszczędność:

   Racjonalne zastosowanie tej technologii pozwala zmniejszyć koszty zużycia energii w budynku, biorąc pod uwagę czynniki takie jak: obecność użytkownika, pora dnia lub faktyczną temperaturę otoczenia.

2. Bezpieczeństwo:

   Wykorzystanie różnego rodzaju czujników, w pełni zintegrowanych, chroni użytkownika przed skutkami nieszczęśliwych zdarzeń typu zalanie i pożar budynku i jego elementów ruchomych, ale również może zapobiec kradzieży mienia.

3. Komfort:

   Inteligentne instalacje pozwalają sterować oświetleniem czy temperaturą bez konieczności przemieszczania się, a przy użyciu pilota (często zaimplementowanego do komputera), co może znacznie wpłynąć na komfort osoby, która z takich instalacji korzysta.

4. Personalizacja:

   Aktualnie większość z inteligentnych rozwiązań jest dostarczana przez producentów sprzętów, na przykład AGD, jednak są one odgórnie narzucone użytkownikowi i nierzadko niemożliwe do personalizacji. Zastosowanie zautomatyzowanej technologii inteligentnych instalacji dostosowanych do użytkownika może pozytywnie wpłynąć na

funkcjonalność urządzeń domowych typu AGD, zwłaszcza uwzględniając czynniki ekonomiczne.

5. Harmonogram:

   Personalizacja systemu jest ściśle związana z harmonogramem jego pracy. Poprzez adaptację systemu do wszystkich użytkowników można efektywnie zarządzać pracą urządzeń w budynku, wcześniej odpowiednio programując ich działanie. Tak usystematyzowana praca pozytywnie wpływa na oszczędność czasu użytkowników.

Zastosowanie inteligentnych technologii w budownictwie staje się standardem. Zwracając uwagę na koszty implementacji tego typu rozwiązań, z powodzeniem można zaaplikować taki system zarządzania w mniejszych obiektach.

4 Raspberry i Arduino

Raspberry Pi (patrz Rys. 4.1) oraz Arduino (patrz Rys. 4.2) są platformami, dzięki którym można wykonać różnego rodzaju systemy automatyki. Pozwalają na podłączenie czujników dostarczających wielu informacji, na podstawie których odbywa się zautomatyzowane sterowanie zasobami systemów.

Raspberry Pi jest pełnowartościowym komputerem z systemem operacyjnym, Arduino Uno jest platformą działającą na mikrokontrolerach. Podstawowym ograniczeniem dla Raspberry Pi (mimo że jest platformą komputerową) jest brak możliwości odczytu danych analogowych bez użycia dodatkowych modułów (potrafi odczytywać jedynie dane cyfrowe). Arduino Uno znakomicie radzi sobie z odczytem danych analogowych i cyfrowych może działać niezależnie, ale jest mniej zaawansowany. Porównanie obu platform przedstawia tabela 4.1:

Tabela 4.1
Porównanie Raspberry pi 3 B+ z Arduino UNO R3

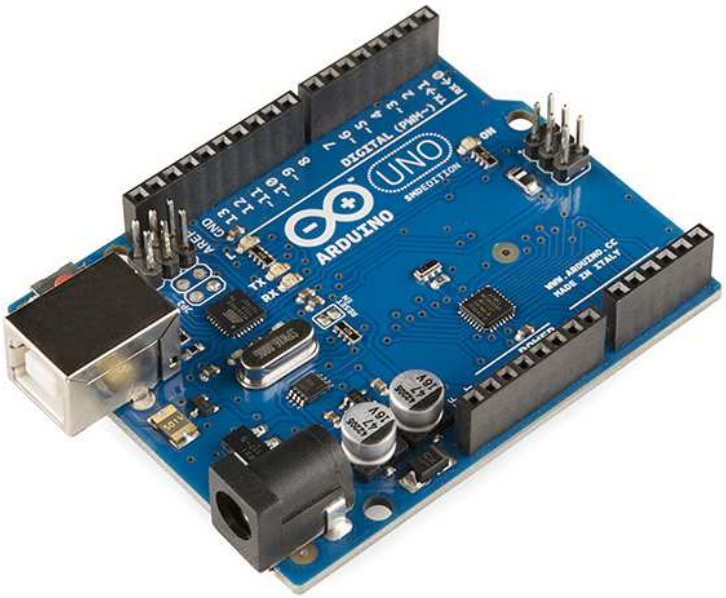
Cechy	**Raspberry pi3 B+**	**Arduino UNO R3**
Wymiary	85 x 56 x 17 mm	75x53x15 mm
Procesor	4 rdzeniowy 1,2 GHz 64-bit	16MHz 8-bit
Pamięć RAM	1024 MB	2 kB
Pamięć FLASH	Do 64 GB (karta micro SD)	32 kB
Zasilanie	5V - micro USB	5V – USB B lub 7-12V zasilacz
Złącza uniwersalne	40 pin (w tym 27 GPIO)	32 pin (w tym 20 GPIO)
Przetwornik analogowy	Brak	6 kanałów
Sieć	10/100 Mbps	Brak
WiFi	802.11 b/g/n 150 Mbps	Brak
Bluetooth	4.1	Brak
USB	4x USB 2.0	1 x USB typ B
Pozostałe interfejsy	I2C, SPI, UART, CSI, DSI	I2C, SPI, UART
Audio / Video	HDMI, jack 3,5 mm	Brak
System operacyjny:	Niezbędny – Linux, Windows, Android, XBMC, inne,	Brak

Źródło: opracowanie własne na podstawie kart technicznych wyżej przedstawionych urządzeń

Rys. 4.1. Raspberry Pi 3 B+

Źródło: https://www.raspberrypi.org/app/uploads/2017/05/Raspberry-Pi-3-Ports-1-1833x1080.jpg (data odczytu:15-12-2017 r.)

Rys. 4.2. Arduino UNO Rev.3



Źródło: https://cdn.sparkfun.com//assets/parts/6/8/1/7/11225-01.jpg (data odczytu:15-12-2017 r.)

Jak pokazano w tabeli 4.1, jedną z nielicznych cech, które łączą oba urządzenia są wymiary. Reszta jest zgoła odmienna, co powoduje, że jedno jest dobrym uzupełnieniem drugiego. W projekcie użyto obu urządzeń (również z powodu kosztów)[2].

[2] Koszt przetwornika ADC współpracującego z I2C to około 60zł, natomiast Arduino UNO około 30zł

Raspberry Pi wspiera wiele systemów operacyjnych, lecz oficjalnym jest Raspbian[3]. Do programowania również można wybierać z mnóstwa języków. W niniejszej pracy autor zdecydował się na język Python ze względu na przejrzystość jego kodu[4]. Funkcjonalnością nie odbiega on od innych języków wysokiego poziomu, jak również posiada rozbudowany pakiet bibliotek. Realizowany w ramach pracy sterownik akwarystyczny obsługiwany będzie z poziomu dowolnej przeglądarki www, za pomocą interface'u wykonanego w PHP w połączeniu z bazą danych (patrz rozdział: 6)

Do programowania Arduino używany jest dedykowany język programowania, bardzo podobny do C/C++[5]. Mikrokontroler zaprogramowano w środowisku ArduinoIDE, którego wygląd przedstawia rysunek 4.3, we wspomnianym pseudo języku C++.

Rys. 4.3. środowisko ArduinoIDE

Źródło: https://pl.wikipedia.org/wiki/Arduino#/media/File:Arduino IDE - v0011 Alpha.png (data odczytu: 12-03-2018 r.)

[3] https://www.raspberrypi.org/downloads/ (data odczytu: 09-03-2018r.)
[4] https://pl.wikipedia.org/wiki/Python (data odczytu: 09-03-2018r.)
[5] C++ jest wieloparadygmatowym językiem programowania ogólnego przeznaczenia. Źródło: https://pl.wikipedia.org/wiki/C++ (data odczytu: 12-03-2018 r.)

5 Elementy składowe sterownika

5.1 Części i elementy pomocnicze

Oprócz samego Raspberry Pi i Arduino UNO oraz kilku różnych czujników, które zostały opisane w późniejszych rozdziałach, w projekcie wykorzystane zostały różnego rodzaju elementy układów elektronicznych, takie jak oporniki czy potencjometry. Bezpośrednimi elementami, które będą załączać poszczególne urządzenia, są przekaźniki.

5.1.1 Przekaźniki

Przekaźniki są to urządzenia stosowane w układach, pozwalające na bezpieczne podłączenie urządzeń. Dla potrzeb niniejszego projektu zostały wykorzystanie 2 moduły z ośmioma przekaźnikami (o maksymalnym dopuszczalnym napięciu przełączania 250V AC, oraz 30 V DC, przy prądzie 10 A), które przy zadanych warunkach na wejściu przekazują prąd do określonego urządzenia (patrz Rys. 5.1.).

Rys. 5.1. moduł z ośmioma przekaźnikami

Źródło: zdjęcie własne

5.1.2 Oporniki i potencjometry

Zarówno opornik, jak i potencjometr to rezystory, które służą do redukcji napięcia w obwodzie elektrycznym. Różnica między nimi polega na możliwości (lub jej braku) regulacji ograniczenia napięcia, który przewodzą. Opornik to bierny element obwodu elektrycznego, zo oznacza, że nie wzmacnia sygnałów elektrycznych, potencjometr natomiast umożliwia regulację natężenia prądu elektrycznego. W budowie sterownika akwariowego wykorzystany został potencjometr liniowy, który charakteryzuje się proporcjonalną zależnością napięcia od prądu.

5.2 Sterowanie oświetleniem – czujnik zmierzchu

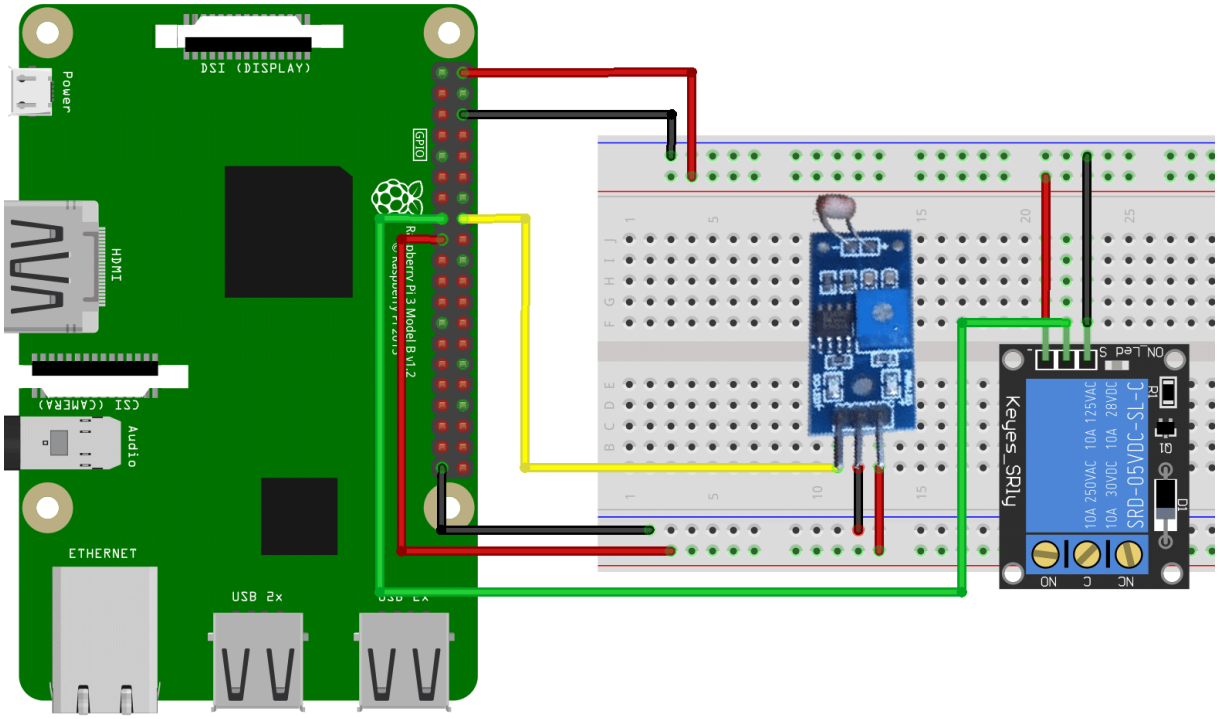
Do sterowania oświetleniem w taki sposób, aby symulować warunki rzeczywiste, użyty został czujnik zmierzchu (patrz Rys. 5.2.). Czujnik ten oparty jest o fotorezystor sprawdzający natężenie światła. Wyposażony jest w potencjometr, dzięki któremu istnieje możliwość regulacji jego czułości, co pozwala uzyskać na wyjściu odpowiednią wartość logiczną.

Rys. 5.2. czujnik zmierzchu

Źródło: zdjęcie własne

Projekt zakłada sterowanie oświetleniem, które składa się z trzech belek oświetleniowych LED. Po zasygnalizowaniu przez czujnik świtała wschodu słońca trzy przekaźniki podają napięcie dla oświetlenia (osobno dla każdej belki). Dodatkowo, w projekcie zastosowano również mechanizm uzupełniający, dzięki któremu można niezależnie od zaprogramowanych parametrów dostosować światło do indywidualnych preferencji.
Schemat podłączenia przedstawia Rys. 5.3. Po odczytaniu zadanej wartości poziomu naświetlenia przekaźnik załącza oświetlenie.

Rys. 5.3. schemat realizacji sterowania oświetleniem



Źródło: opracowanie własne (za pomocą programu fritzing)

Kod realizujący sterowanie oświetleniem przedstawia się następująco:

Listing 5.1: kod obsługujący pracę czujnika natężenia światła

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      #wejscia
      GPIO.setup(23, GPIO.IN) #wejcie z czujnika swiatla

      #wyjscia
      GPIO.setup(17, GPIO.OUT) #przekaznik LED_1
      GPIO.setup(27, GPIO.OUT) #przekaznik LED_2
      GPIO.setup(22, GPIO.OUT) #przekaznik LED_3
      GPIO.output(17, 1) #przekaznik LED_1
      GPIO.output(27, 1) #przekaznik LED_2
      GPIO.output(22, 1) #przekaznik LED_3

      #zmienne globalne
```

```
      stan_LED_1 = 0
      stan_LED_2 = 0
      stan_LED_3 = 0

      man_auto_LED_1 = 0
      man_auto_LED_2 = 0
      man_auto_LED_3 = 0

def zapytaj_db():#aby nirobic ciaglych zapytan do bazy

      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")

      #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      a=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]

      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT stan FROM stany_urzadzen" )  #WHERE Lp=4"
      #print "pierwszy SELECT";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            a[i]=dane
            i=i+1

      global stan_LED_1
      global stan_LED_2
      global stan_LED_3

      stan_LED_1 = a[0]
      stan_LED_2 = a[1]
      stan_LED_3 = a[2]

      #sprawdzenie man / auto urzadzen
      #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      b=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT man_auto FROM stany_urzadzen" )
      #print "man / auto??";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            b[i]=dane
            i=i+1

      global man_auto_LED_1
      global man_auto_LED_2
      global man_auto_LED_3

      man_auto_LED_1 = b[0]
      man_auto_LED_2 = b[1]
      man_auto_LED_3 = b[2]

      db.close()
      return

ustawienia()
```

```
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")
    #zmien wartosc w bazie i stowrz plik
      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      #po zmianie jesli nie ma pliku to go tworz
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")
      db.close()


def czujnik_swiatla():
      dev_ON = GPIO.LOW       #mod przekaznikow sterowany stanem niskim
      dev_OFF= GPIO.HIGH      #co oznacza ze przekaznik zalacza urzadzenie
przy GPIO.LOW
      czujnik_natezenia_swiatla = GPIO.input(23)
      print "czujnik swiatla: "
      print czujnik_natezenia_swiatla
      if man_auto_LED_1==0: #LEDY - jesli 0 ustawienia manualne
            if stan_LED_1==0:
                  GPIO.output(17, dev_OFF)
            else:
                  GPIO.output(17, dev_ON)
            if stan_LED_2==0:
                  GPIO.output(27, dev_OFF)
            else:
                  GPIO.output(27, dev_ON)
            if stan_LED_3==0:
                  GPIO.output(22, dev_OFF)
            else:
                  GPIO.output(22, dev_ON)

      else:             #dzialanie automatyczne
            if czujnik_natezenia_swiatla==0:
                  GPIO.output(17, dev_ON) #LED1 on
                  GPIO.output(27, dev_ON) #LED1 on
                  GPIO.output(22, dev_ON) #LED1 on
                  if stan_LED_1==0:
                        update_stanu_w_bazie(1, 1) #Lp dla LED_1 = 1
                  if stan_LED_2==0:
                        update_stanu_w_bazie(1, 2) #Lp dla LED_2 = 2
                  if stan_LED_3==0:
                        update_stanu_w_bazie(1, 3) #Lp dla LED_3 = 3
            else:
                  GPIO.output(17, dev_OFF)
                  if stan_LED_1==1:
                        update_stanu_w_bazie(0, 1)
                  GPIO.output(27, dev_OFF)
                  if stan_LED_2==1:
                        update_stanu_w_bazie(0, 2)
                  GPIO.output(22, dev_OFF)
                  if stan_LED_3==1:
                        update_stanu_w_bazie(0, 3)
```

```
while True:
      global delay
      #os.system("clear")

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
            czujnik_swiatla()
```

Źródło: opracowanie własne

Niezależne sterowanie oświetleniem jest możliwe dzięki odpowiedniemu zapisowi statusu oświetlenia (auto / manual) w bazie danych (patrz rozdział: 6). Decyzja dotycząca automatycznego czy też manualnego sterowania oświetleniem jest wyborem użytkownika i może być zmieniana z poziomu przeglądarki. Poniższy kod realizuje sterowanie oświetleniem poprzez stronę www:

Listing 5.2: kod wyświetlający ustawienia oświetlenia

```
<?php
$plik = file('private/accessdb.txt');
      //zczytanie pliku (do tablicy po linii)
$servername = chop($plik[0]); //chop() usuwa znaki konca linii
$username = chop($plik[1]);
$password = chop($plik[2]);
$dbname = chop($plik[3]);

$conn = new mysqli($servername, $username, $password, $dbname);
 $sql = "SELECT stan FROM stany_urzadzen";
 $result = $conn->query($sql);
 for ($stan=array(); $row=$result->fetch_assoc(); $stan[]=$row[stan]);

 echo '<font color="white" size="1">';

$stan_LED_1 = $stan[0]; echo $stan_LED_1;
$stan_LED_2 = $stan[1]; echo $stan_LED_2;
$stan_LED_3 = $stan[2]; echo $stan_LED_3;
  echo '<br>';
 $sql = "SELECT man_auto FROM stany_urzadzen";
 $result = $conn->query($sql);
 for ($man_auto=array(); $row=$result->fetch_assoc();
$man_auto[]=$row[man_auto]);

$man_auto_LED_1 = $man_auto[0]; echo $man_auto_LED_1;
$man_auto_LED_2 = $man_auto[1]; echo $man_auto_LED_2;
$man_auto_LED_3 = $man_auto[2]; echo $man_auto_LED_3;
```

```
 echo '</font>';



//zmiana ustawien oswietlenia
$stan__LED_1=$_GET['stan__LED_1'];
$stan__LED_2=$_GET['stan__LED_2'];
$stan__LED_3=$_GET['stan__LED_3'];
$man_auto__LED_1=$_GET['man_auto__LED_1'];
if(isset($_GET['stan__LED_1'])){
      $zmien_stan__LED_1 = "UPDATE stany_urzadzen SET stan='" .
$stan__LED_1 . "' WHERE Lp='1'";
            if($conn->query($zmien_stan__LED_1) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: sterownik.php");
                  } else {echo 'Error: tu' .  $conn->error;}
}
if(isset($_GET['stan__LED_2'])){
      $zmien_stan__LED_2 = "UPDATE stany_urzadzen SET stan='" .
$stan__LED_2 . "' WHERE Lp='2'";
            if($conn->query($zmien_stan__LED_2) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: sterownik.php");
                  } else {echo 'Error: tu' .  $conn->error;}
}
if(isset($_GET['stan__LED_3'])){
      $zmien_stan__LED_3 = "UPDATE stany_urzadzen SET stan='" .
$stan__LED_3 . "' WHERE Lp='3'";
            if($conn->query($zmien_stan__LED_3) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: sterownik.php");
                  } else {echo 'Error: tu' .  $conn->error;}
}
if(isset($_GET['man_auto__LED_1'])){
      $zmien_tryb__LED_1 = "UPDATE stany_urzadzen SET man_auto='"
                        . $man_auto__LED_1 . "' WHERE Lp='1'";
            if($conn->query($zmien_tryb__LED_1) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: sterownik.php");
                  } else {echo 'Error: lub tuuuu ' .  $conn->error;}
}

echo '<center>
            <table border="0">
            <tr><td colspan="8">
            <center><h1>Sterownik akwariowy</h1></center></td></tr>
      echo '<tr><td colspan="7"></td></tr>

            <tr><td colspan="7"><br><b>Oswietlenie: </b></td></tr>
            <tr>
            <td rowspan="2"></td>
            <td rowspan="2"></td>
            <td colspan="2" align="center">Tryb pracy: </td>


            <td align="center">Belka 1</td>
            <td align="center">Belka 2</td>
            <td align="center">Belka 3</td>
            </tr>
            <tr>
```

```
            <td colspan="2" align="center">';
                  if($man_auto_LED_1==1)
                        {echo '<a href="sterownik.php?man_auto__LED_1=0">
                   <img src="images/auto.png"></a>';}
                        else {echo '<a
href="sterownik.php?man_auto__LED_1=1">
                    <img src="images/man.png"></a>';}
            echo '</td>
            <td align="center" height="50">';
               if($stan_LED_1==1){
                        echo '<a href="sterownik.php?stan__LED_1=0">
                  <img src="images/on.png"></a>';     }
                        else {      echo '<a
href="sterownik.php?stan__LED_1=1">
                   <img src="images/off.png"></a>';}

            echo '</td>
            <td align="center">';
               if($stan_LED_2==1){
                        echo '<a href="sterownik.php?stan__LED_2=0">
                  <img src="images/on.png"></a>';     }
                        else {      echo '<a
href="sterownik.php?stan__LED_2=1">
                          <img src="images/off.png"></a>';}

            echo '</td>
            <td align="center">';
               if($stan_LED_3==1){
        echo '<a href="sterownik.php?stan__LED_3=0">
                  <img src="images/on.png"></a>';     }
        else {    echo '<a href="sterownik.php?stan__LED_3=1">
                 <img src="images/off.png"></a>';}

            echo '</td>
            </tr>
      </table>';
?>
```

Źródło: opracowanie własne

5.3 Sterowanie temperaturą – czujnik temperatury (grzałka, lodówka)

W celu sterowania temperaturą w akwarium użyto wodoodpornego czujnika temperatury DS18B20 z interfejsem 1-wire przedstawionego na rysunku 5.4. Czujnik temperatury oprócz wskazań aktualnych wartości służy do obsługi grzałki oraz lodówki, gdzie wykorzystywany jest jako termostat.

Rys. 5.4. czujnik temperatury

Źródło: zdjęcie własne

Za komunikację z interfejsem 1-wire odpowiadają moduły jądra w1-gpio oraz w1-therm. Ich wymogiem jest podłączenie 1-wire do pinu GPIO4. Zczytywanie bezpośrednio wartości z GPIO zwraca wartość *1*, gdy jest pomiar, lub *0*, gdy go nie ma. W katalogu /sys/bus/w1/devices znajdują się podkatalogi o nazwach w postaci 28-xxx, gdzie xxx to unikatowy identyfikator czujnika. W podkatalogu 28-03168bc29aff znajduje się plik w1_slave, gdzie zostaje zapisywana aktualna temperatura.

Aby czujnik ładował się automatycznie, konieczna jest modyfikacja w postaci dodania na końcu pliku /etc/modules trzech wierszy:

```
wire
w1-gpio
w1-therm
```

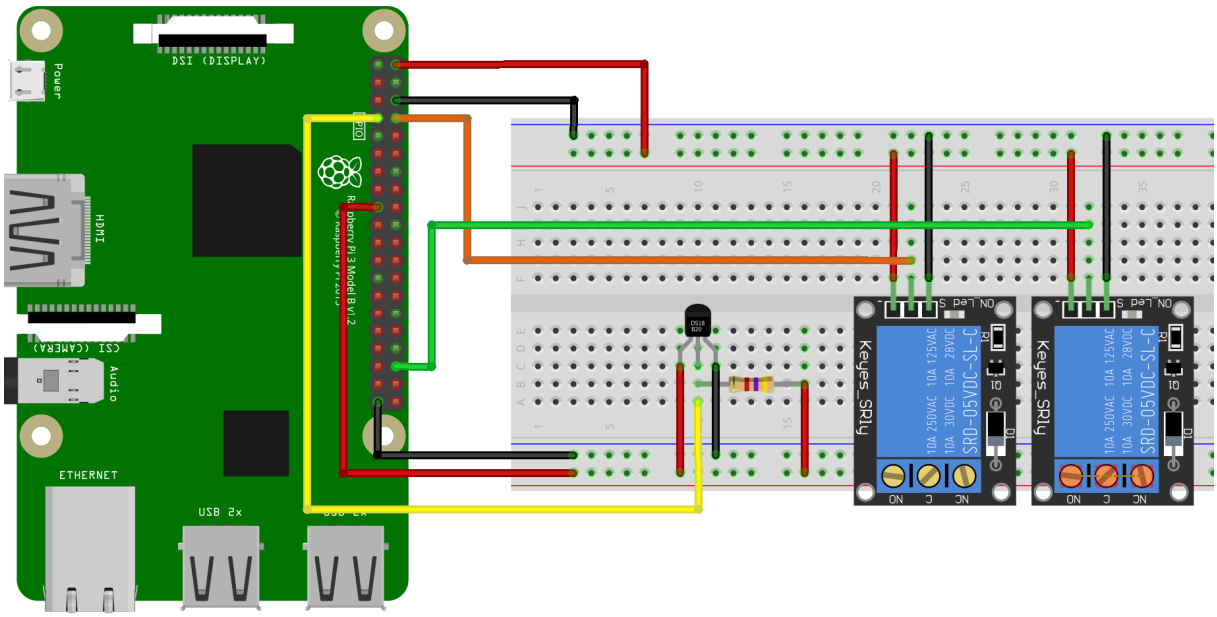
natomiast w pliku config.txt, który jest odpowiedzialny za ładowanie modułów

```
dtoverlay=w1-gpio
gpiopin=4
```

dane przekazywane są do funkcji realizującej termostat.

Schemat poglądowy połączeń przedstawia rysunek 5.5.

Rys. 5.5. schemat realizacji sterowania temperaturą w akwarium



Źródło: Opracowanie własne (za pomocą programu fritzing)

Pomiędzy pinem sygnałowym czujnika a napięciem należy zastosować opornik 4,7 kΩ. Jeden z przekaźników obsługuje włączanie oraz wyłącznie grzałki, drugi lodówki. Całość została oprogramowana w Pythonie:

Listing 5.3: kod obsługujący pracę grzałki oraz lodówki na podstawie danych z czujnika temperatury

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      #wejscia
      GPIO.setup(4, GPIO.IN) #wejcie z termometru

      #wyjscia
      GPIO.setup(14, GPIO.OUT) #przekaznik grzalki
      GPIO.setup(16, GPIO.OUT) #przekaznik lodowki
```

```
      GPIO.output(14, 1) #przekaznik grzalki
      GPIO.output(16, 1) #przekaznik lodowki

      #zmienne globalne
      stan_grzalka = 0
      stan_lodowka = 0
      man_auto_grzalka = 0
      man_auto_lodowka = 0

      zadana_temperatura = 0

def zapytaj_db():#aby nirobic ciaglych zapytan do bazy

      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")

    #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      a=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]

      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT stan FROM stany_urzadzen" )  #WHERE Lp=4"
      #print "pierwszy SELECT";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            a[i]=dane
            i=i+1

      global stan_grzalka
      global stan_lodowka
      stan_grzalka = a[3]
      stan_lodowka = a[4]

      #sprawdzenie man / auto urzadzen
      #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      b=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT man_auto FROM stany_urzadzen" )
      #print "man / auto??";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            b[i]=dane
            i=i+1

      global man_auto_grzalka
      global man_auto_lodowka
      man_auto_grzalka = b[3]
      man_auto_lodowka = b[4]

      #sprawdzenie zadanej temperatury
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT wartosc FROM Temperatura" )
      print "pytam o temperature ";
```

```
      for row in cur.fetchall():
            temperatura=row[0]
            print temperatura

      global zadana_temperatura
      zadana_temperatura=temperatura


      db.close()
      return

ustawienia()
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")
      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")

      db.close()

def sterowanie_temperatura():
      dev_ON = GPIO.LOW       #mod. przekaznikow sterowany stanem niskim
      dev_OFF= GPIO.HIGH      #co oznacza ze przekaznik zalacza urzadzenie
przy GPIO.LOW
      # 1-wire interface
      file = open('/sys/bus/w1/devices/28-03168bc29aff/w1_slave')
      filecontent = file.read()
      file.close()
      #Temperatura
      stringvalue = filecontent.split("\n")[1].split(" ")[9]
                        #wycigniecie z pliku samej wartosci temperatury
      temperatura = float(stringvalue[2:]) / 1000
                        #bardziej cywizliowany spos wyswietlania
      print "Aktualna temperatura: "
      print temperatura
      punkt_wlaczenia_lodowki = zadana_temperatura+2 #tolerancja
bezczynnosci
      print "Lodowka wylaczy sie po przekroczeniu: "
      print punkt_wlaczenia_lodowki

      if man_auto_grzalka==0: #jesli 0 ustawienia manualne
            if stan_grzalka==0:
                  GPIO.output(14, dev_OFF)
            else:
                  GPIO.output(14, dev_ON)
      else:             #dzialanie automatyczne
            if temperatura<zadana_temperatura:
                  GPIO.output(14, dev_ON) #grzalka on
                  if stan_grzalka==0:
                        update_stanu_w_bazie(1, 4)
            else:
                  GPIO.output(14, dev_OFF) #grzalka off
```

```
                  if stan_grzalka==1:
                        update_stanu_w_bazie(0, 4)


      if man_auto_lodowka==0: #jesli 0 ustawienia manualne
            if stan_lodowka==0:
                  GPIO.output(16, dev_OFF)
            else:
                  GPIO.output(16, dev_ON)
      else:             #dzialanie automatyczne
            if temperatura>punkt_wlaczenia_lodowki:
                  GPIO.output(16, dev_ON) #lodowka on
                  if stan_lodowka==0:
                        update_stanu_w_bazie(1, 5)
            else:
                  GPIO.output(16, dev_OFF) #lodowka off
                  if stan_lodowka==1:
                        update_stanu_w_bazie(0, 5)

      return


while True:
      global delay
      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
            sterowanie_temperatura()
```

Źródło: opracowanie własne

Wartość zadanej temperatury znajduje się w bazie danych (patrz rozdział: 6), którą można modyfikować z poziomu przeglądarki www. Poniżej znajduje się fragment kodu, pozwalający na modyfikację wartości temperatury w bazie danych:

Listing 5.4: kod obsługujący modyfikację wartości temperatury w bazie danych

```
<?php
$plik = file('private/accessdb.txt');
      //zczytanie pliku (do tablicy po linii)
$servername = chop($plik[0]); //chop() usuwa znaki konca linii
$username = chop($plik[1]);
$password = chop($plik[2]);
$dbname = chop($plik[3]);

echo '<center><h1>Zmiana temperatury</h1><br>
         Aktualnie termostat ustawiony jest na: <b>';
$conn = new mysqli($servername, $username, $password, $dbname);
```

```
 $sql = "SELECT wartosc FROM Temperatura";
 $result = $conn->query($sql);
 if($result->num_rows > 0){
       while($wiersz = $result->fetch_assoc()) {
                   echo $wiersz["wartosc"];
             }
      }
       else {
             echo 'BLAD';
      }

echo ' C </b> ';

echo ' <form method="post" name="formularz"
            action="zmiana_temperatury.php">
            Aby przestawic wpisz wartosc:
            <input size="5" type="text"  name="wartosc"><b> C </b>
            <input type="submit" name="zmien_temp" value="Zmien"></form>';

$wartosc = $_POST['wartosc'];
$zmien_temp = $_POST['zmien_temp'];

if(isset($zmien_temp)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}


$sql1 = "UPDATE Temperatura SET wartosc='$wartosc' WHERE Lp='1'";
if($conn->query($sql1) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: sterownik.php");
} else {echo 'Error:2 ' .  $conn->error;}

}
$conn->close();
echo '<a href="sterownik.php"><button> << powrot << </button></a>';
?>
```

Źródło: opracowanie własne

5.4 Kontrolowanie PH poprzez sterowanie dawkowaniem CO2 – sonda PH

W akwarystyce w celu przyśpieszenia wzrostu roślin stosuje się nawożenie dwutlenkiem węgla. CO_2 trzeba kontrolować, gdyż obniża on poziom PH wody, co z kolei może być szkodliwe dla ryb. Kontrola PH sprowadza się do odpowiedniego sterowania elektrozaworem na dopływie CO_2 oraz napowietrzaczem. Do odczytu PH użyto sondy PH przedstawionej na rysunku 5.7 oraz Arduino (patrz rozdział: 4) wraz wyświetlaczem LCD 2x16 znaków przedstawionym na rysunku 5.6. W celu poprawnego działania wyświetlacza należy zastosować potencjometr (patrz rozdział: 5.1.2) w celu korekcji kontrastu wyświetlanych informacji. Zastosowany wyłącznik daje możliwość wygaszenia ekranu bez konieczności wyłączania urządzenia.

Rys. 5.6. wyświetlacz (zdjęcie poglądowe)

Źródło: https://ae01.alicdn.com/kf/HTB1BO3sIVXXXXc9XFXXq6xXFXXXY.jpg
(data odczytu: 18.12.2017 r.)

Sonda przekazuje dane analogowe w postaci liniowej. Aby wskazania były poprawne, należy wyniki przeliczyć na *mV*. W teorii idealna sonda w temperaturze 25°C powinna przekazać wartości zgodne z tabelą 5.1. W rzeczywistości wartości odbiegają od idealnych, dlatego sondę należy kalibrować. W tym przypadku należy użyć dwóch różnych buforów z określonym PH cieczy w celu wyznaczenia równania prostej przechodzącej przez dwa punkty[6]:

$$A = (x_A, y_A) \text{ oraz } B = (x_B, y_B)$$

gdzie:

[6] https://www.matemaks.pl/rownanie-prostej-przechodzacej-przez-dwa-punkty.html (data odczytu: 29-11-2017r.)

punkt *A* - jest pierwszym pomiarem, dla którego x_A to wartość pierwszego buforu PH, y_A to odczyt wartości sondy zanurzonej w cieczy,

punkt *B* - jest drugim pomiarem, dla którego x_B to wartość drugiego buforu PH, y_B to odczyt wartości sondy zanurzonej w cieczy.

Wyznaczone punkty należy podstawić do wzoru:

$$(y - y_A)(x_B - x_A) - (y_B - y_A)(x - x_A)$$

i przekształcić równanie do postaci

$$y=a*x+b$$

W ten sposób uzyskamy właściwe wartości poziomu PH wody w akwarium. Schemat połączeń sondy PH z Arduino przedstawia rysunek 5.8.

Tabela 5.1
wartości napięcia dla określonego PH

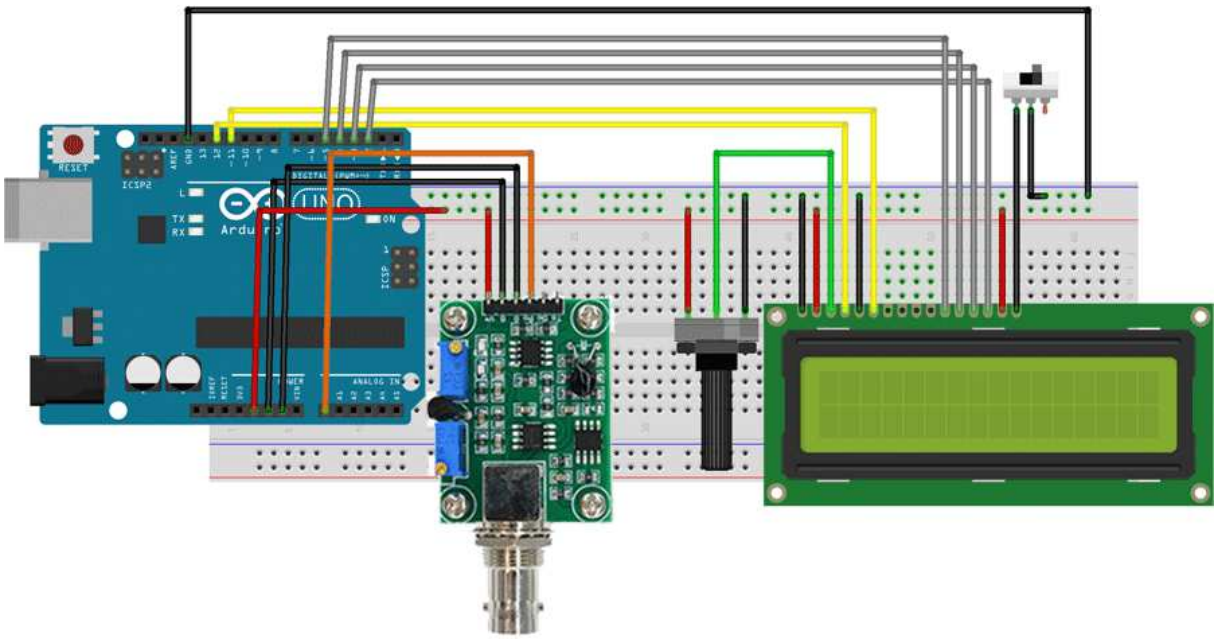
napięcie (*mV*)	**wartość PH**	**napięcie (*mV*)**	**wartość PH**
414,12	0,00	414,12	14,00
354,96	1,00	354,96	13,00
295,80	2,00	295,80	12,00
236,64	3,00	236,64	11,00
177,48	4,00	177,48	10,00
118,32	5,00	118,32	9,00
59,16	6,00	59,16	8,00
0,00	7,00	0,00	7,00

Źródło: https://www.dfrobot.com/wiki/index.php/PH meter(SKU: SEN0161)
(data odczytu: 29-11-2017r.)

Rys. 5.7. sonda PH

Źródło: zdjęcie własne

Rys. 5.8. schemat realizacji odczytu PH wody



Źródło: opracowanie własne (za pomocą programu fritzing oraz Gimp)

Całość została oprogramowana w języku pseudo C, który jest wykorzystywany do pisania szkiców Arduino w środowisku Arduino IDE:

Listing 5.5: kod obsługujący pracę sondy która sprawdza PH wody

```
#include <LiquidCrystal.h>
// Arduino Modul do pomiaru PH

const int pin = A0;
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  Serial.begin(9600);
}

void loop() {
  float tablica[10];
  float wyniki;
  float wynik_koncowy_poprzedni = 0;
for (int k = 0; k < 9; k++) {
  int tab[10];
  int postep;

  unsigned long int sredn_wynik = 0;
  // 10 probek do tablicy z 10 ms
  for (int i = 0; i < 10; i++) {
    tab[i] = analogRead(pin);
    delay(10);
  }
```

```
  // sortowanie wynikow w tablicy
  for (int i = 0; i < 9; i++) {
    for (int j = i + 1; j < 10; j++) {
      if (tab[i] > tab[j]) {
        postep = tab[i];
        tab[i] = tab[j];
        tab[j] = postep;
      }
    }
  }
  // odrzucenie skrajnych wynikow
  // tak dla lepszej dokladnosci
  for (int i = 2; i < 8; i++) {
    sredn_wynik += tab[i];
  }
  // obliczenie sredniej wartosci ph
  //  0-14 pH
  float srednPH = (float)sredn_wynik * 5.0 / 1024 / 6; //tu mV

  //float kolejnePH = -2.7745 * srednPH + 15.0348;// czwarty z wynikow OK
  //float kolejnePH = -5.45 * srednPH + 21.45;// kalibraacja 2.10.2017
pomiedzy 4.00 a 9,18
  //float kolejnePH = -11.05 * srednPH + 36.90;//  7.10.2017 pomiedzy 6.86
a 9,18
 //  float kolejnePH = -5.6078 * srednPH + 23.6835;//25.01.2018 prawie
ideal (pomiedzy 4 a 6,86)
  float kolejnePH = -5.6586 * srednPH + 23.8356; //25.01.2018 idealnie
(pomiedzy 6,86 a 9,18):D
  // wswietlenie wynikow
  //Serial.print("Zmierzone pH: ");
  tablica[k] = kolejnePH;
 delay(10);
}

for (int i = 0; i < 9; i++) {
    for (int j = i + 1; j < 10; j++) {
      if (tablica[i] > tablica[j]) {
        wyniki = tablica[i];
        tablica[i] = tablica[j];
        tablica[j] = wyniki;
      }
    }
  }
 float wynik_koncowy = tablica[4];
 float tekst = float(wynik_koncowy);
  Serial.println(wynik_koncowy);
  //Serial.println(tekst);
  lcd.begin(16,2);
  lcd.print("Ph: ");
  lcd.setCursor(1,5);
  lcd.print(tekst);
  // pauza 900 ms
  //char msgBuffer[5];
  //String tekst = String(wynik_koncowy,4);
  //tekst.println(wynik_koncowy);
  delay(10);
}
```

Źródło: opracowanie własne

Taka realizacja pozwala na używanie urządzenia odczytującego poziom PH wody niezależnie od całego sterownika. Oczywiście w takim przypadku należy zastosować dodatkowy zasilacz, który obsłuży Arduino. W przypadku podłączenia do sterownika urządzenie przesyła dane przez USB i też przez USB jest zasilane.
Poniższy kod przedstawia sposób odczytu danych z Arduino przez Raspberry Pi, przez co możliwe jest sterowaniem PH wody:

Listing 5.6: kod pozwalający na odczyt danych z Arduino i sterowanie PH

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      #wyjscia
      GPIO.setup(5, GPIO.OUT) #przekaznik 230V_1
      GPIO.setup(6, GPIO.OUT) #przekaznik 230V_2
      GPIO.output(5, 1) #przekaznik 230V_1
      GPIO.output(6, 1) #przekaznik 230V_2

      stan_230V_1 = 0 #napowietrzacz
      stan_230V_2 = 0 #co2
      man_auto_230V_1 = 0
      man_auto_230V_2 = 0

      zadane_ph = 0

def zapytaj_db():#aby nirobic ciaglych zapytan do bazy

      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")

    #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      a=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]

      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT stan FROM stany_urzadzen" )  #WHERE Lp=4"
      #print "pierwszy SELECT";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            a[i]=dane
            i=i+1
```

```
      global stan_230V_1
      global stan_230V_2
      stan_230V_1 = a[5]
      stan_230V_2 = a[6]

      #sprawdzenie man / auto urzadzen
      #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      b=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT man_auto FROM stany_urzadzen" )
      #print "man / auto??";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            b[i]=dane
            i=i+1

      global man_auto_230V_1
      global man_auto_230V_2
      man_auto_230V_1 = b[5]
      man_auto_230V_2 = b[6]

      #sprawdzenie zadanego PH
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT wartosc FROM PH" )
      print "pytam o zadane PH ";

      for row in cur.fetchall():
            ph=row[0]
            print ph

      global zadane_ph
      zadane_ph=ph

      db.close()
      return
ustawienia()
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")

      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")
      db.close()

def sterowanie_ph():
      dev_ON = GPIO.LOW       #mod przekaznikow sterowany stanem niskim
      dev_OFF= GPIO.HIGH      #co oznacza ze przekaznik zalacza urzadzenie
przy GPIO.LOW
      import serial
      ser = serial.Serial('/dev/ttyACM0', 9600)
```

```
      #while 1: #
                                                                  ty zrobic while
ser == True
      #odczyt_kontrlony = ser.readline()
      #print ser.read()
      aktualne_ph = float(ser.readline())
      #aktualne_ph = 7.33
      print "aktualne ph: "
      print aktualne_ph
      ustw_zakres_ph = float(zadane_ph)
      ph_min = ustw_zakres_ph-0.05
      ph_max = ustw_zakres_ph+0.05
      #ph_min = 7.33-0.05
      #ph_max = 7.33+0.05

      if man_auto_230V_2==0: #co2 - jesli 0 ustawienia manualne
            if stan_230V_2==0:
                  GPIO.output(6, dev_OFF)
            else:
                  GPIO.output(6, dev_ON)
      else:             #dzialanie automatyczne
            if aktualne_ph>ph_max:
                  GPIO.output(6, dev_ON) #co2 on
                  if stan_230V_2==0:
                        update_stanu_w_bazie(1, 7) #Lp dla co2 = 7
            else:
                  GPIO.output(6, dev_OFF)
                  if stan_230V_2==1:
                        update_stanu_w_bazie(0, 7)

      if man_auto_230V_1==0: #napowietrzacz - jesli 0 ustawienia manualne
            if stan_230V_1==0:
                  GPIO.output(5, dev_OFF)
            else:
                  GPIO.output(5, dev_ON)
      else:             #dzialanie automatyczne
            if aktualne_ph<ph_min:
                  GPIO.output(5, dev_ON) #napowietrzacz on
                  if stan_230V_1==0:
                        update_stanu_w_bazie(1, 6) #Lp dla napowietrzacza=6
            else:
                  GPIO.output(5, dev_OFF)
                  if stan_230V_1==1:
                        update_stanu_w_bazie(0, 6)
      return

while True:
      global delay

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"
      else:
            print "nie ma pliku"
            sterowanie_ph()
```

Źródło: opracowanie własne

Podobnie jak w przypadku ustawień temperatury, zadaną wartość PH wody można modyfikować przez stronę www:

Listing 5.7: kod obsługujący modyfikację wartości PH

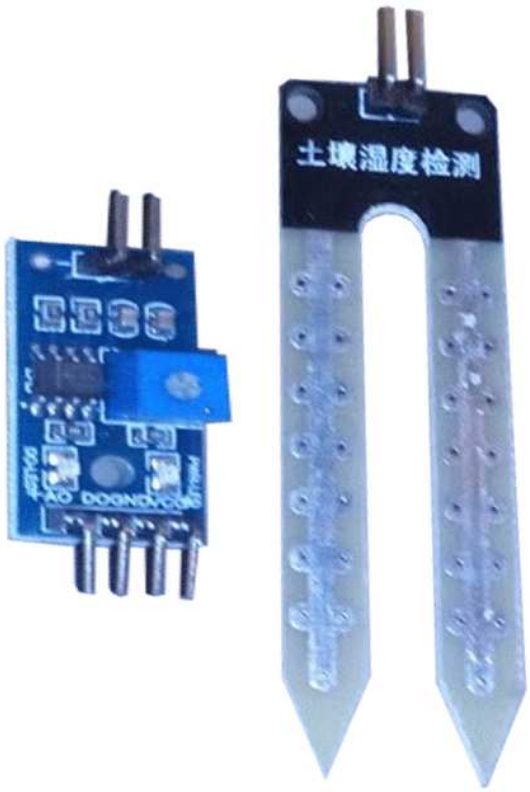
```
<?php
$plik = file('private/accessdb.txt');
      //zczytanie pliku (do tablicy po linii)
$servername = chop($plik[0]); //chop() usuwa znaki konca linii
$username = chop($plik[1]);
$password = chop($plik[2]);
$dbname = chop($plik[3]);

echo '<center><h1>Zmiana ustawienie PH wody</h1><br>
         Aktualnie PH ustawione jest na: <b>';
$conn = new mysqli($servername, $username, $password, $dbname);

 $sql = "SELECT wartosc FROM PH";
 $result = $conn->query($sql);
 if($result->num_rows > 0){
       while($wiersz = $result->fetch_assoc()) {
                   echo $wiersz["wartosc"];
             }
      }
       else {
             echo 'BLAD';
      }

echo '</b> ';
echo ' <form method="post" name="formularz"
            action="zmiana_PH.php">
            Aby przestawic wpisz zadana wartosc:
            <input size="5" type="text"  name="wartosc">
            <input type="submit" name="zmien_ph" value="Zmien"></form>';

$wartosc = $_POST['wartosc'];
$zmien_ph = $_POST['zmien_ph'];

if(isset($zmien_ph)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql1 = "UPDATE PH SET wartosc='$wartosc' WHERE Lp='1'";
if($conn->query($sql1) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: sterownik.php");
} else {echo 'Error:2 ' .  $conn->error;}

}
$conn->close();

echo '<a href="sterownik.php"><button> << powrot << </button></a>';
?>
```

Źródło: opracowanie własne

5.5 Alarm powodziowy – czujnik zawilgocenia

Dzięki czujnikowi poziomu zawilgocenia gleby, przedstawionemu na zdjęciu 5.9, można wykonać system szybkiego ostrzegania przed zalaniem. Czujnik posiada dwa wyjścia - analogowe i cyfrowe. Dzięki wyjściu analogowemu można uzyskać dokładny poziom zawilgocenia gleby (wykorzystywany np. w systemie automatycznego nawadniania roślin).
W projekcie skorzystano z wyjścia cyfrowego, gdyż oczekiwaną (ale niepożądaną) informacją na wyjściu jest stan logiczny o pojawieniu się wilgoci. Czujnik zostanie zainstalowany pod szafką, na której stoi zbiornik, w rejonie filtra zewnętrznego (to właśnie miejsce jest najczęstszym, gdzie pojawiają się przecieki).

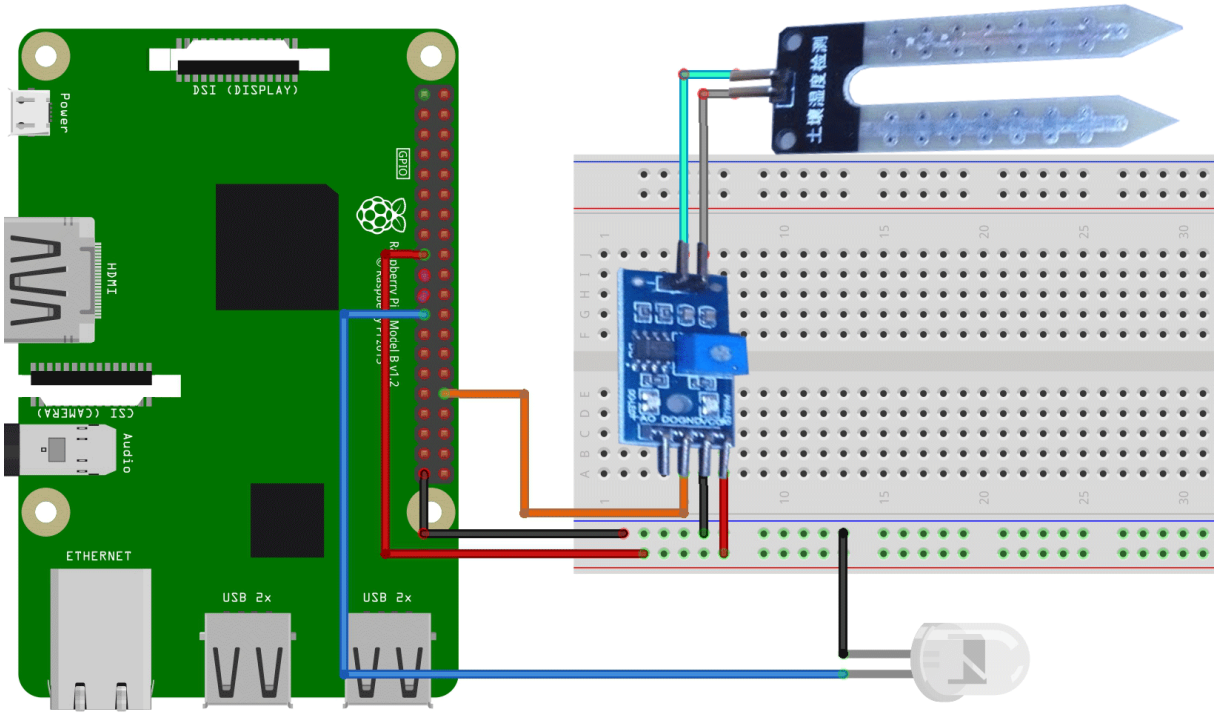
Rys. 5.9. czujnik poziomu zawilgocenia



Źródło: zdjęcie własne

Schemat podłączenia oraz działania czujnika jest bardzo podobny do działania czujnika poziomu oświetlenia opisanego w rozdziale 5.2. Po odczytaniu stanu zawilgocenia zostaje uruchomiony sygnał ostrzegawczy w postaci migającej diody ostrzegawczej.

Rys. 5.10. schemat realizacji systemu ostrzegania przed zalaniem



Źródło: opracowanie własne (za pomocą programu fritzing)

Listing 5.8: kod obsługujący pracę czujnika wilgotności

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      #wejscia
      GPIO.setup(12, GPIO.IN) #wejcie z czujnika wilgotn.

      #wyjscia
      GPIO.setup(11, GPIO.OUT) #dioda alarmowa
      GPIO.output(11, 0) #diada alarm

def alarm():
      #wylacz wszystko!!!
      GPIO.output(14, 1) #przekaznik grzalki
      GPIO.output(16, 1) #przekaznik lodowki
      GPIO.output(2, 1) #przekaznik karmika
      GPIO.output(3, 1) #przekaznik 231V_6
      GPIO.output(17, 1) #przekaznik LED_1
```

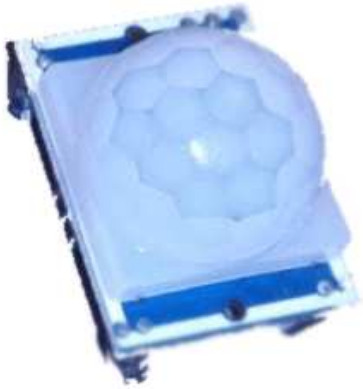
```
      GPIO.output(27, 1) #przekaznik LED_2
      GPIO.output(22, 1) #przekaznik LED_3
      GPIO.output(5, 1) #przekaznik 230V_1
      GPIO.output(6, 1) #przekaznik 230V_2
      GPIO.output(13, 1) #przekaznik 230V_3
      GPIO.output(19, 1) #przekaznik 230V_4
      GPIO.output(26, 1) #przekaznik 230V_5
      GPIO.output(24, 1) #przekaznik pump_1
      GPIO.output(25, 1) #przekaznik pump_2
      GPIO.output(20, 1) #przekaznik pump_3
      GPIO.output(21, 1) #przekaznik pump_4
      #migaj na alarm

      while 1:
            GPIO.output(11, 1)
            time.sleep(1)
            GPIO.output(11, 0)
            time.sleep(1)

def czujnik_wilg():
      czujnik_zalania = GPIO.input(12)
      print "czujnik zalania: "
      print czujnik_zalania
      if czujnik_zalania==0:
            print "A L A R M ! ! !"
            alarm()
            #GPIO.output(11, dev_ON) #dioda on
      else:
            GPIO.output(11, 0)

while True:
      global delay
      #os.system("clear")

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
            czujnik_wilg()
```

Źródło: opracowanie własne

5.6 Sterowanie ozdobami – piaskospad – czujnik ruchu

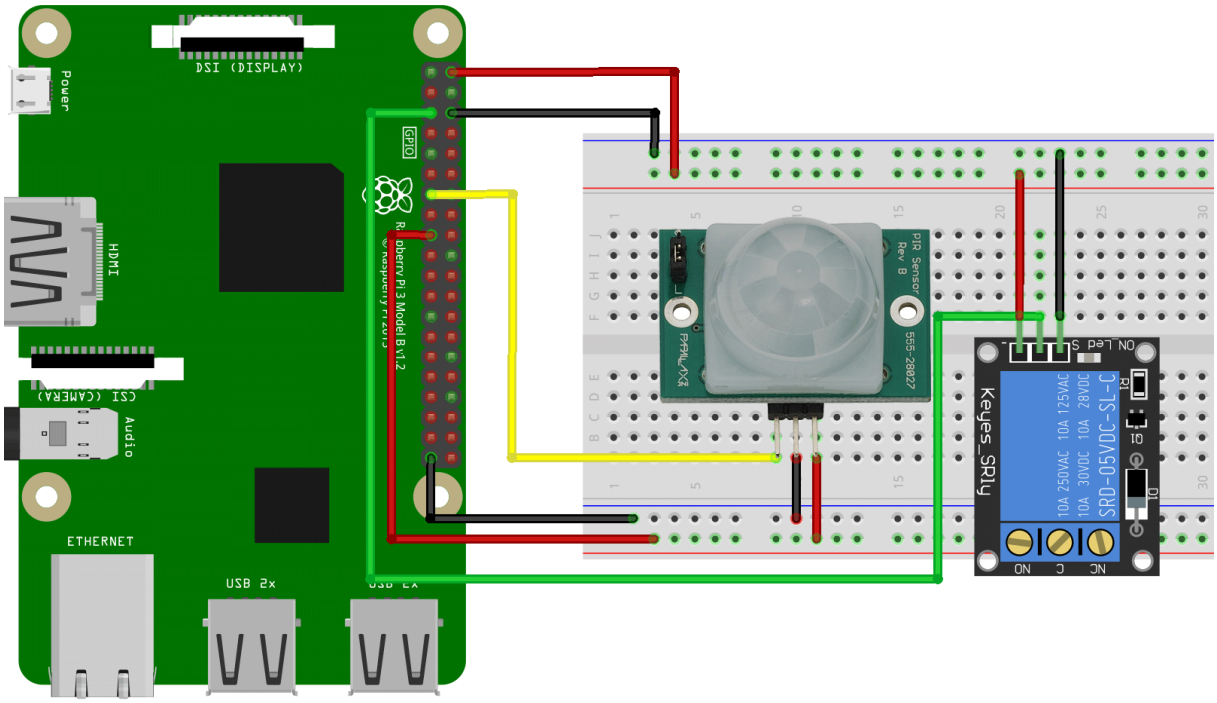
Akwarium to niewątpliwie element dekoracyjny w mieszkaniu. Do jego wystroju stosuje się wiele ozdób. Niektóre wymagają zasilania, ale też nie jest konieczne, aby były włączone przez cały czas, szczególnie gdy nie ma nikogo w pobliżu. Aby zrealizować powyższe działanie, zostanie wykorzystany czujnik ruchu PIR przedstawiony na zdjęciu 5.11.

Rys. 5.11. czujnik ruchu PIR



Źródło: zdjęcie własne

Rys. 5.12. schemat realizacji sterowanie ozdobami



Źródło: opracowanie własne (za pomocą programu fritzing)

Projekt zakłada, że po wykryciu ruchu przez czujnik przekaźnik włączy działanie ozdoby (w tym przypadku piaskospadu) na określony czas - 10 min.

Schemat poglądowy połączeń przedstawia rysunek 5.12.

Program umożliwiający prawidłowe działanie czujnika ruchu wraz z przekaźnikiem:

Listing 5.9: kod obsługujący czujnik ruchu

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      #wejscia
      GPIO.setup(18, GPIO.IN) #wejcie z czujnika ruchu
      GPIO.setup(26, GPIO.OUT) #przekaznik 230V_5
      GPIO.output(26, 1) #przekaznik 230V_5
      stan_230V_5 = 0 #piaskospad
      man_auto_230V_5 = 0

def zapytaj_db():
      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")

    #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      a=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]

      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT stan FROM stany_urzadzen" )
      #print "pierwszy SELECT";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            a[i]=dane
            i=i+1
      global stan_230V_5
      stan_230V_5 = a[9]

      #sprawdzenie man / auto urzadzen
      #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      b=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT man_auto FROM stany_urzadzen" )
      #print "man / auto??";
      i=0
      for row in cur.fetchall():
            dane=row[0]
```

```
            #print dane
            b[i]=dane
            i=i+1

      global man_auto_230V_5
      man_auto_230V_5 = b[9]

      # odczyt czasowych ustawien
      c=[0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT godz_on FROM czasowe" )
      i=0
      for row in cur.fetchall():
            dane=row[0]
            c[i]=dane
            i=i+1
      global ON_piaskospad
      ON_piaskospad = c[3]

      c=[0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT godz_off FROM czasowe" )
      i=0
      for row in cur.fetchall():
            dane=row[0]
            c[i]=dane
            i=i+1
      global OFF_piaskospad
      OFF_piaskospad = c[3]

      db.close()
      return

ustawienia()
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                        user="jakobe",
                        passwd=„tajnehaslo”,
                        db="akwarium")
    #zmien wartosc w bazie i stowrz plik
      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      #cur.execute("UPDATE stany_urzadzen SET stan=1 WHERE Lp=4" )
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      #po zmianie jesli nie ma pliku to go tworz
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")

      db.close()

def pir():
      dev_ON = GPIO.LOW       #mod przekaznikow sterowany stanem niskim
      dev_OFF= GPIO.HIGH      #co oznacza ze przekaznik zalacza urzadzenie
przy GPIO.LOW
      czujnik_PIR = GPIO.input(18)

      print "czujnik ruchu: "
```

```
      print czujnik_PIR
      if man_auto_230V_5==0: #piaskospad - jesli 0 ustawienia manualne
            if stan_230V_5==0:
                  GPIO.output(26, dev_OFF)
            else:
                  GPIO.output(26, dev_ON)
      else:             #dzialanie automatyczne
            if czujnik_PIR==1:

                  czas = time.strftime('%H:%M:%S')
                  now = datetime.datetime.now()
                  now_plus_10 = now + datetime.timedelta(minutes = 10)
                  czas_plus_10 = now_plus_10.strftime('%H:%M:%S')

                  db = MySQLdb.connect(host="localhost",
                              user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")
                  cur = db.cursor()
                  cur.execute("UPDATE czasowe SET godz_on='%s',
                              godz_off='%s' WHERE Lp=4" % (czas,
czas_plus_10))
                  db.commit()
                  if not os.path.exists("change.txt"):
                        os.mknod("change.txt")
                  db.close()
                  zapytaj_db()
                  #teraz = now
                  #now_plus_10 = now + datetime.timedelta(minutes = 1)

      now = time.strftime("%H:%M:%S") #+ "\n"
      teraz = str(now) #aby mozna bylo porownac
      ONpiaskospad = str(ON_piaskospad)
      OFFpiaskospad = str(OFF_piaskospad)

      #aby mozna bylo porownac (bez przeliczania jest prblem przy
      #czasie gdy np godzina jest jedna cyfra)
      teraz = str(now)
      (h, m, s)= teraz.split(':')
      teraz = int(h)*3600+int(m)*60+int(s)

      ONpiaskospad  = str(ON_piaskospad )
      (h, m, s)= ONpiaskospad .split(':')
      ONpiaskospad  = int(h)*3600+int(m)*60+int(s)

      OFFpiaskospad  = str(OFF_piaskospad )
      (h, m, s)= OFFpiaskospad .split(':')
      OFFpiaskospad  = int(h)*3600+int(m)*60+int(s)

      if ((teraz > ONpiaskospad) and (teraz < OFFpiaskospad)):
            print "piaskospad ON"
            GPIO.output(26, dev_ON) #piskospad on
            if stan_230V_5==0:
                  update_stanu_w_bazie(1, 10) #Lp dla piaskospad = 10
      else:
            GPIO.output(26, dev_OFF)
            if stan_230V_5==1:
                  update_stanu_w_bazie(0, 10)


while True:
```

```
      global delay
      #os.system("clear")

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
            pir()
```

Źródło: opracowanie własne

Możliwe jest również manualne sterowanie piaskospadem, dzięki odpowiedniemu zapisowi statusu oświetlenia (auto / manual) w bazie danych. Poniższy kod wyświetla sposób sterowania ozdobą oraz możliwość zmiany:

Listing 5.10: kod wyświetlający sposób sterowania ozdobą oraz możliwość zmiany

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      #wejscia
      GPIO.setup(18, GPIO.IN) #wejcie z czujnika ruchu
      GPIO.setup(26, GPIO.OUT) #przekaznik 230V_5
      GPIO.output(26, 1) #przekaznik 230V_5
      stan_230V_5 = 0 #piaskospad
      man_auto_230V_5 = 0

def zapytaj_db():
      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")

    #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      a=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
```

```
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT stan FROM stany_urzadzen" )
      #print "pierwszy SELECT";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            a[i]=dane
            i=i+1
      global stan_230V_5
      stan_230V_5 = a[9]

      #sprawdzenie man / auto urzadzen
      #  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
      b=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT man_auto FROM stany_urzadzen" )
      #print "man / auto??";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            b[i]=dane
            i=i+1

      global man_auto_230V_5
      man_auto_230V_5 = b[9]

      # odczyt czasowych ustawien
      c=[0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT godz_on FROM czasowe" )
      i=0
      for row in cur.fetchall():
            dane=row[0]
            c[i]=dane
            i=i+1
      global ON_piaskospad
      ON_piaskospad = c[3]

      c=[0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT godz_off FROM czasowe" )
      i=0
      for row in cur.fetchall():
            dane=row[0]
            c[i]=dane
            i=i+1
      global OFF_piaskospad
      OFF_piaskospad = c[3]

      db.close()
      return

ustawienia()
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                           user="jakobe",
                           passwd=„tajnehaslo”,
```

```
                    db="akwarium")
    #zmien wartosc w bazie i stowrz plik
      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      #cur.execute("UPDATE stany_urzadzen SET stan=1 WHERE Lp=4" )
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      #po zmianie jesli nie ma pliku to go tworz
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")

      db.close()

def pir():
      dev_ON = GPIO.LOW       #mod przekaznikow sterowany stanem niskim
      dev_OFF= GPIO.HIGH      #co oznacza ze przekaznik zalacza urzadzenie
przy GPIO.LOW
      czujnik_PIR = GPIO.input(18)

      print "czujnik ruchu: "
      print czujnik_PIR
      if man_auto_230V_5==0: #piaskospad - jesli 0 ustawienia manualne
            if stan_230V_5==0:
                  GPIO.output(26, dev_OFF)
            else:
                  GPIO.output(26, dev_ON)
      else:             #dzialanie automatyczne
            if czujnik_PIR==1:

                  czas = time.strftime('%H:%M:%S')
                  now = datetime.datetime.now()
                  now_plus_10 = now + datetime.timedelta(minutes = 10)
                  czas_plus_10 = now_plus_10.strftime('%H:%M:%S')

                  db = MySQLdb.connect(host="localhost",
                              user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")
                  cur = db.cursor()
                  cur.execute("UPDATE czasowe SET godz_on='%s',
                              godz_off='%s' WHERE Lp=4" % (czas,
czas_plus_10))
                  db.commit()
                  if not os.path.exists("change.txt"):
                        os.mknod("change.txt")
                  db.close()
                  zapytaj_db()
                  #teraz = now
                  #now_plus_10 = now + datetime.timedelta(minutes = 1)

      now = time.strftime("%H:%M:%S") #+ "\n"
      teraz = str(now) #aby mozna bylo porownac
      ONpiaskospad = str(ON_piaskospad)
      OFFpiaskospad = str(OFF_piaskospad)

      #aby mozna bylo porownac (bez przeliczania jest prblem przy
      #czasie gdy np godzina jest jedna cyfra)
      teraz = str(now)
      (h, m, s)= teraz.split(':')
      teraz = int(h)*3600+int(m)*60+int(s)
```

```
      ONpiaskospad  = str(ON_piaskospad )
      (h, m, s)= ONpiaskospad .split(':')
      ONpiaskospad  = int(h)*3600+int(m)*60+int(s)

      OFFpiaskospad  = str(OFF_piaskospad )
      (h, m, s)= OFFpiaskospad .split(':')
      OFFpiaskospad  = int(h)*3600+int(m)*60+int(s)

      if ((teraz > ONpiaskospad) and (teraz < OFFpiaskospad)):
            print "piaskospad ON"
            GPIO.output(26, dev_ON) #piskospad on
            if stan_230V_5==0:
                  update_stanu_w_bazie(1, 10) #Lp dla piaskospad = 10
      else:
            GPIO.output(26, dev_OFF)
            if stan_230V_5==1:
                  update_stanu_w_bazie(0, 10)


while True:
      global delay
      #os.system("clear")

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
            pir()
```

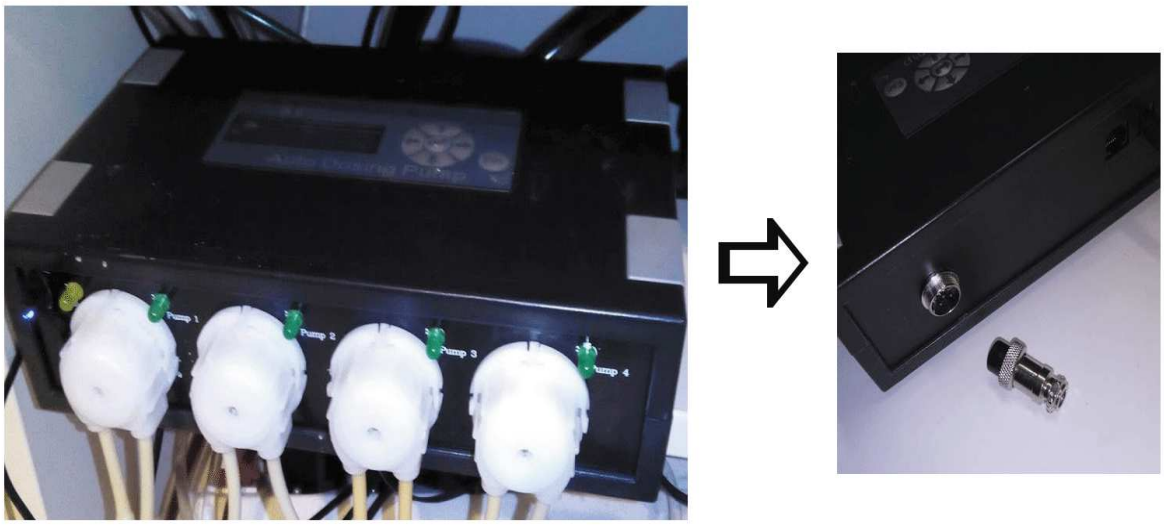
Źródło: opracowanie własne

5.7 Sterowanie pompami dozującymi nawozy

W celu możliwości sterowania nawożeniem zbiornika wykorzystano gotowe urządzenie sterujące dozowaniem płynów wyposażone w cztery pompy, które są w stanie podawać płyny z dokładnością do 1ml firmy JEBAO model DP-4[7]. Urządzenie zmodyfikowano poprzez wyprowadzenia pinów z urządzenia "na zewnątrz", dzięki czemu możliwe jest jego podłączenie i sterowanie poprzez Raspberry Pi (Rys. 5.13).

[7] https://pl.aliexpress.com/item/Jebao-Auto-Dosing-Peristalsis-Pump-DP-4-For-Coral-Reef-Aquarium-4-Pumps-Head-Jebao-DP/32581473095.html?spm=a2g0s.9042311.0.0.z2AL4x (data odczytu: 30-11-2017r.)

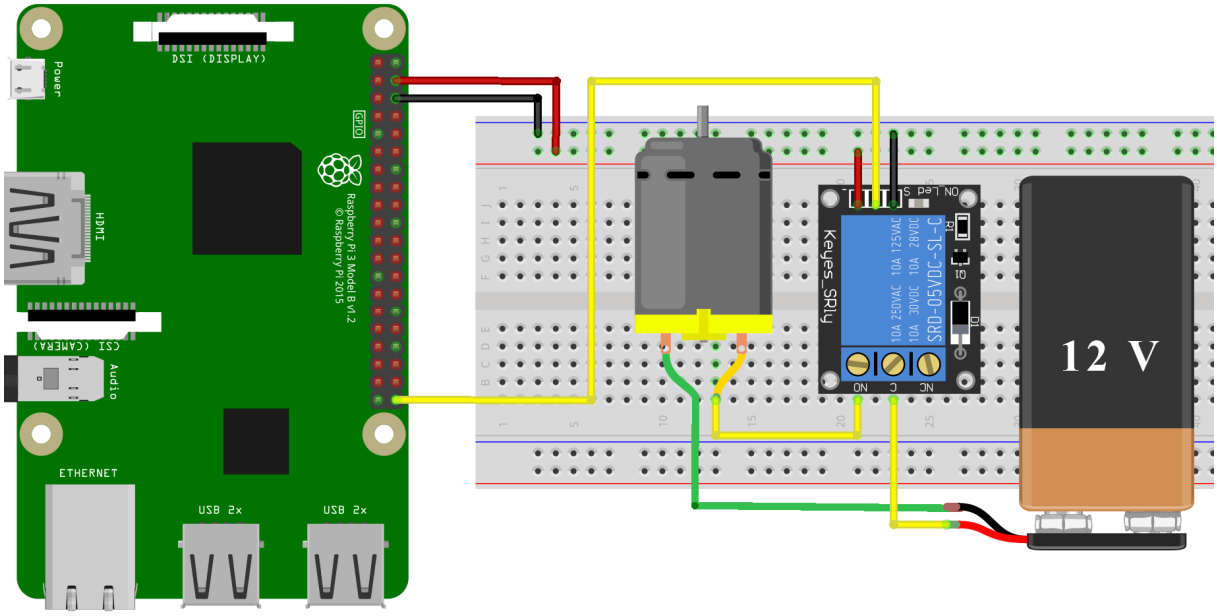
Rys. 5.13. pompa JEBAO DP-4 oraz wykonana przeróbka



Źródło: zdjęcia własne (obróbka za pomocą programu Gimp)

Schemat podłączenia pojedynczej pompy przedstawia rysunek 5.14.

Rys. 5.14. schemat realizacji sterowania pompą



Źródło: opracowanie własne (za pomocą programu fritzing oraz Gimp)

Poniższy kod przedstawia pracę pomp dozujących, które podają zadane ilości płynu w odpowiednich godzinach. Dane te przechowywane są w bazie danych:

Listing 5.11: kod realizujący działanie pomp dozujących

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      #wyjscia
      GPIO.setup(24, GPIO.OUT) #przekaznik pump_1
      GPIO.setup(25, GPIO.OUT) #przekaznik pump_2
      GPIO.setup(20, GPIO.OUT) #przekaznik pump_3
      GPIO.setup(21, GPIO.OUT) #przekaznik pump_4
      GPIO.output(24, GPIO.HIGH) #przekaznik pump_1
      GPIO.output(25, GPIO.HIGH) #przekaznik pump_2
      GPIO.output(20, GPIO.HIGH) #przekaznik pump_3
      GPIO.output(21, GPIO.HIGH) #przekaznik pump_4
      on_off_pump_1 = 0 #pompy
      godzON_pump_1 = 0
      Ile_dawek_pump_1 = 0

      on_off_pump_2 = 0
      godzON_pump_2 = 0
      Ile_dawek_pump_2 = 0

      on_off_pump_3 = 0
      godzON_pump_3 = 0
      Ile_dawek_pump_3 = 0

      on_off_pump_4 = 0
      godzON_pump_4 = 0
      Ile_dawek_pump_4 = 0

      znak_pump_1 = 0
      znak_pump_2 = 0
      znak_pump_3 = 0
      znak_pump_4 = 0


def zapytaj_db():#aby nirobic ciaglych zapytan do bazy

      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")

      #sprawdzenie ustawien pomp
      #  0,1,2,3 odczyt godzin wlaczania
      b=[0,0,0,0]
```

```
cur = db.cursor() #create a Cursor object.
cur.execute("SELECT godzON FROM PUMPS" )
i=0
for row in cur.fetchall():
      dane=row[0]
      #print dane
      b[i]=dane
      i=i+1
global godzON_pump_1
global godzON_pump_2
global godzON_pump_3
global godzON_pump_4

godzON_pump_1 = b[0]
godzON_pump_2 = b[1]
godzON_pump_3 = b[2]
godzON_pump_4 = b[3]

#  0,1,2,3 odczyt ile ml ma dozowac
b=[0,0,0,0]
cur = db.cursor() #create a Cursor object.
cur.execute("SELECT Ile_dawek FROM PUMPS" )
i=0
for row in cur.fetchall():
      dane=row[0]
      #print dane
      b[i]=dane
      i=i+1
global Ile_dawek_pump_1
global Ile_dawek_pump_2
global Ile_dawek_pump_3
global Ile_dawek_pump_4

Ile_dawek_pump_1 = b[0]
Ile_dawek_pump_2 = b[1]
Ile_dawek_pump_3 = b[2]
Ile_dawek_pump_4 = b[3]


#  0,1,2,3 odczyt czy aktywna
b=[0,0,0,0]
cur = db.cursor() #create a Cursor object.
cur.execute("SELECT on_off FROM PUMPS" )
i=0
for row in cur.fetchall():
      dane=row[0]
      #print dane
      b[i]=dane
      i=i+1
global on_off_pump_1
global on_off_pump_2
global on_off_pump_3
global on_off_pump_4

on_off_pump_1 = b[0]
on_off_pump_2 = b[1]
on_off_pump_3 = b[2]
on_off_pump_4 = b[3]

#sprawdzenie zadanych dawek
cur = db.cursor() #create a Cursor object.
```

```
      cur.execute("SELECT czas_dawki FROM Dawki" )
      print "pytam o zadane czasy dawek ";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            print dane
            a[i]=dane
            i=i+1

      global czas_dawki_pomp
      global czas_dawki_karmienia

      czas_dawki_pomp = a[0]
      czas_dawki_karmienia = a[1]

      db.close()
      return

ustawienia()
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")
    #zmien wartosc w bazie i stowrz plik
      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      #cur.execute("UPDATE stany_urzadzen SET stan=1 WHERE Lp=4" )
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      #po zmianie jesli nie ma pliku to go tworz
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")

      db.close()


def pompy():
      dev_ON = GPIO.LOW       #mod przekaznikow sterowany stanem niskim
      dev_OFF= GPIO.HIGH      #co oznacza ze przekaznik zalacza urzadzenie
przy GPIO.LOW
      now = time.strftime("%H:%M:%S") #+ "\n"
      teraz = str(now) #aby mozna bylo porownac

      if on_off_pump_1==0:  #pompa 1
            GPIO.output(24, dev_OFF)
      else:

            znak_pump_1 = godzON_pump_1 + datetime.timedelta(0,5) #+ 5 sek
         #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp
            ON_pump_1 = str(godzON_pump_1)
            znacznik_pump_1 = str(znak_pump_1)
            if ((teraz > ON_pump_1) and (teraz < znacznik_pump_1)):
                  print "dawkowanie - popmpa 1 podaje plyn"
                  GPIO.output(24, dev_ON)
                  dawkowanie = Ile_dawek_pump_1 * czas_dawki_pomp
                  time.sleep(dawkowanie)
                  GPIO.output(24, dev_OFF)
```

```
      else:
            GPIO.output(24, dev_OFF)


if on_off_pump_2==0: #pompa 2
      GPIO.output(25, dev_OFF)
else:

      znak_pump_2 = godzON_pump_2 + datetime.timedelta(0,5) #+ 5 sek
   #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp
      ON_pump_2 = str(godzON_pump_2)
      znacznik_pump_2 = str(znak_pump_2)
      if ((teraz > ON_pump_2) and (teraz < znacznik_pump_2)):
            print "dawkowanie - popmpa 2 podaje plyn"
            GPIO.output(25, dev_ON)
            dawkowanie = Ile_dawek_pump_2 * czas_dawki_pomp
            time.sleep(dawkowanie)
            GPIO.output(25, dev_OFF)

      else:
            GPIO.output(25, dev_OFF)

if on_off_pump_3==0: #pompa 3
      GPIO.output(20, dev_OFF)
else:

      znak_pump_3 = godzON_pump_3 + datetime.timedelta(0,5) #+ 5 sek
   #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp

      ON_pump_3 = str(godzON_pump_3)
      znacznik_pump_3 = str(znak_pump_3)
      if ((teraz > ON_pump_3) and (teraz < znacznik_pump_3)):
            print "dawkowanie - popmpa 3 podaje plyn"
            GPIO.output(20, dev_ON)
            dawkowanie = Ile_dawek_pump_3 * czas_dawki_pomp
            time.sleep(dawkowanie)
            GPIO.output(20, dev_OFF)

      else:
            GPIO.output(20, dev_OFF)

if on_off_pump_4==0: #pompa 4
      GPIO.output(21, dev_OFF)
else:

      znak_pump_4 = godzON_pump_4 + datetime.timedelta(0,5) #+ 5 sek
   #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp

      ON_pump_4 = str(godzON_pump_4)
      znacznik_pump_4 = str(znak_pump_4)
      if ((teraz > ON_pump_4) and (teraz < znacznik_pump_4)):
            print "dawkowanie - popmpa 4 podaje plyn"
            GPIO.output(21, dev_ON)
            dawkowanie = Ile_dawek_pump_4 * czas_dawki_pomp
            time.sleep(dawkowanie)
            GPIO.output(21, dev_OFF)

      else:
            GPIO.output(21, dev_OFF)
```

```
while True:
      global delay
      #os.system("clear")

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
            pompy()
```

Źródło: opracowanie własne

Możliwość zmiany ustawień pomp realizowane jest z poziomu przeglądarki internetowej. Odpowiedni kod w PHP modyfikuje zawartość bazy:

Listing 5.12: kod modyfikujący ustawienia pomp dozujących

```
<?php
$plik = file('private/accessdb.txt');
      //zczytanie pliku (do tablicy po linii)
$servername = chop($plik[0]); //chop() usuwa znaki konca linii
$username = chop($plik[1]);
$password = chop($plik[2]);
$dbname = chop($plik[3]);
echo '<center><h1>USTAWIENIA POMP:</h1>';
//odczytanie ktore pompy sa aktywne
$conn = new mysqli($servername, $username, $password, $dbname);
 $sql = "SELECT on_off FROM PUMPS";
 $result = $conn->query($sql);
 for ($on_off=array(); $row=$result->fetch_assoc();
$on_off[]=$row[on_off]);

$stan_pump_1 = $on_off[0]; //echo $stan_pump_1;
$stan_pump_2 = $on_off[1];
$stan_pump_3 = $on_off[2];
$stan_pump_4 = $on_off[3];
//odczytanie godzin o ktorch pompy sie aktywuja
 $sql = "SELECT godzON FROM PUMPS";
 $result = $conn->query($sql);
 for ($godzON=array(); $row=$result->fetch_assoc();
$godzON[]=$row[godzON]);
$godzON_pump_1 = $godzON[0];
 $godzON_pump_2 = $godzON[1];
 $godzON_pump_3 = $godzON[2];
 $godzON_pump_4 = $godzON[3];

//odczytanie dawek poszczegolnych pomp
 $sql = "SELECT Ile_dawek FROM PUMPS";
 $result = $conn->query($sql);
 for ($Ile_dawek=array(); $row=$result->fetch_assoc();
$Ile_dawek[]=$row[Ile_dawek]);
```

```
 $Ile_dawek_pump_1 = $Ile_dawek[0];
 $Ile_dawek_pump_2 = $Ile_dawek[1];
 $Ile_dawek_pump_3 = $Ile_dawek[2];
 $Ile_dawek_pump_4 = $Ile_dawek[3];

//zmiana ustawien pomp
$stan__pump_1=$_GET['stan__pump_1'];
$stan__pump_2=$_GET['stan__pump_2'];
$stan__pump_3=$_GET['stan__pump_3'];
$stan__pump_4=$_GET['stan__pump_4'];

if(isset($_GET['stan__pump_1'])){
      $zmien_stan__pump_1 = "UPDATE PUMPS SET on_off='" . $stan__pump_1 .
"' WHERE Lp='1'";
            if($conn->query($zmien_stan__pump_1) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_pomp.php");
                  } else {echo 'Error' .  $conn->error;}
}
if(isset($_GET['stan__pump_2'])){
      $zmien_stan__pump_2 = "UPDATE PUMPS SET on_off='" . $stan__pump_2 .
"' WHERE Lp='2'";
            if($conn->query($zmien_stan__pump_2) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_pomp.php");
                  } else {echo 'Error' .  $conn->error;}
}
if(isset($_GET['stan__pump_3'])){
      $zmien_stan__pump_3 = "UPDATE PUMPS SET on_off='" . $stan__pump_3 .
"' WHERE Lp='3'";
            if($conn->query($zmien_stan__pump_3) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_pomp.php");
                  } else {echo 'Error' .  $conn->error;}
}
if(isset($_GET['stan__pump_4'])){
      $zmien_stan__pump_4 = "UPDATE PUMPS SET on_off='" . $stan__pump_4 .
"' WHERE Lp='4'";
            if($conn->query($zmien_stan__pump_4) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_pomp.php");
                  } else {echo 'Error' .  $conn->error;}
}
echo '<center><table border="1">'; //tabela pompy 1
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Pompa Nr 1: ';
            if($stan_pump_1==1){
                        echo '<a href="ustawienia_pomp.php?stan__pump_1=0">
                         <img src="images/on.png"> </a>';   }
                        else {      echo '<a
href="ustawienia_pomp.php?stan__pump_1=1">
                          <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
            <br>Pompa Nr 1 wlacza sie o godzinie: <b>' . $godzON_pump_1 .
'</b> i podaje
            jednorazowo <b>' . $Ile_dawek_pump_1 . ' ml </b>plynu <br>';
echo '</center></td></tr>
            <tr><td align="center">';
//ustawienia godziny dla pompy 1
```

```
echo ' <form method="post" name="pompa1"
            action="ustawienia_pomp.php">
            Zmien godzine pompy nr 1 (wpisz wartosc w formacie HH:MM:SS)
            <input size="5" type="text"  name="godzina_pump_1">
            <input type="submit" name="zmien_godzina_pump_1"
value="Ustaw"></form>';
$godzina_pump_1 = $_POST['godzina_pump_1'];
$zmien_godzina_pump_1 = $_POST['zmien_godzina_pump_1'];
if(isset($zmien_godzina_pump_1)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql1 = "UPDATE PUMPS SET godzON='$godzina_pump_1' WHERE Lp='1'";
if($conn->query($sql1) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:2 ' .  $conn->error;}}

//ustawienia ilosci ml dla pompy 1
echo ' <form method="post" name="dawka_pompa1"
            action="ustawienia_pomp.php">
            Zmien dawke dla pompy nr 1 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_pump_1">  ml
            <input type="submit" name="zmien_dawka_pump_1"
value="Ustaw"></form>';

$dawka_pump_1 = $_POST['dawka_pump_1'];
$zmien_dawka_pump_1 = $_POST['zmien_dawka_pump_1'];

if(isset($zmien_dawka_pump_1)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}
$sql1 = "UPDATE PUMPS SET Ile_dawek='$dawka_pump_1' WHERE Lp='1'";
if($conn->query($sql1) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:1 ' .  $conn->error;}

}
echo '</td></tr></table>';
echo '</table>';//tabela glowna
echo '<br>';
//koniec pompy 1

echo '<center><table border="1">'; //tabela pompy 2
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Pompa Nr 2: ';
            if($stan_pump_2==1){
                        echo '<a href="ustawienia_pomp.php?stan__pump_2=0">
                         <img src="images/on.png"> </a>';   }
                        else {      echo '<a
href="ustawienia_pomp.php?stan__pump_2=1">
                          <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
            <br>Pompa Nr 2 wlacza sie o godzinie: <b>' . $godzON_pump_2 .
'</b> i podaje
            jednorazowo <b>' . $Ile_dawek_pump_2 . ' ml </b>plynu <br>';

echo '</center></td></tr>
            <tr><td align="center">';
```

```
//ustawienia godziny dla pompy 2
echo ' <form method="post" name="pompa2"
            action="ustawienia_pomp.php">
            Zmien godzine pompy nr 2 (wpisz wartosc w formacie HH:MM:SS)
            <input size="5" type="text"  name="godzina_pump_2">
            <input type="submit" name="zmien_godzina_pump_2"
value="Ustaw"></form>';

$godzina_pump_2 = $_POST['godzina_pump_2'];
$zmien_godzina_pump_2 = $_POST['zmien_godzina_pump_2'];
if(isset($zmien_godzina_pump_2)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql2 = "UPDATE PUMPS SET godzON='$godzina_pump_2' WHERE Lp='2'";
if($conn->query($sql2) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:2 ' .  $conn->error;}}

//ustawienia ilosci ml dla pompy 1
echo ' <form method="post" name="dawka_pompa2"
            action="ustawienia_pomp.php">
            Zmien dawke dla pompy nr 2 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_pump_2">  ml
            <input type="submit" name="zmien_dawka_pump_2"
value="Ustaw"></form>';

$dawka_pump_2 = $_POST['dawka_pump_2'];
$zmien_dawka_pump_2 = $_POST['zmien_dawka_pump_2'];

if(isset($zmien_dawka_pump_2)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql2 = "UPDATE PUMPS SET Ile_dawek='$dawka_pump_2' WHERE Lp='2'";
if($conn->query($sql2) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:2 ' .  $conn->error;}

}
echo '</td></tr></table>';
echo '</table>';//tabela glowna
echo '<br>';
//koniec pompy 2

echo '<center><table border="1">'; //tabela pompy 3
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Pompa Nr 3: ';
            if($stan_pump_3==1){
                        echo '<a href="ustawienia_pomp.php?stan__pump_3=0">
                         <img src="images/on.png"> </a>';   }
                        else {      echo '<a
href="ustawienia_pomp.php?stan__pump_3=1">
                        <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
            <br>Pompa Nr 3 wlacza sie o godzinie: <b>' . $godzON_pump_3 .
'</b> i podaje
            jednorazowo <b>' . $Ile_dawek_pump_3 . ' ml </b>plynu <br>';
```

```
echo '</center></td></tr>
            <tr><td align="center">';

//ustawienia godziny dla pompy 3
echo ' <form method="post" name="pompa3"
            action="ustawienia_pomp.php">
            Zmien godzine pompy nr 3 (wpisz wartosc w formacie HH:MM:SS)
            <input size="5" type="text"  name="godzina_pump_3">
            <input type="submit" name="zmien_godzina_pump_3"
value="Ustaw"></form>';

$godzina_pump_3 = $_POST['godzina_pump_3'];
$zmien_godzina_pump_3 = $_POST['zmien_godzina_pump_3'];
if(isset($zmien_godzina_pump_3)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql3 = "UPDATE PUMPS SET godzON='$godzina_pump_3' WHERE Lp='3'";
if($conn->query($sql3) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:3 ' .  $conn->error;}}

//ustawienia ilosci ml dla pompy 3
echo ' <form method="post" name="dawka_pompa3"
            action="ustawienia_pomp.php">
            Zmien dawke dla pompy nr 3 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_pump_3">  ml
            <input type="submit" name="zmien_dawka_pump_3"
value="Ustaw"></form>';
$dawka_pump_3 = $_POST['dawka_pump_3'];
$zmien_dawka_pump_3 = $_POST['zmien_dawka_pump_3'];

if(isset($zmien_dawka_pump_3)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql3 = "UPDATE PUMPS SET Ile_dawek='$dawka_pump_3' WHERE Lp='3'";
if($conn->query($sql3) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:3 ' .  $conn->error;}

}
echo '</td></tr></table>';
echo '</table>';//tabela glowna
echo '<br>';
//koniec pompy 3

echo '<center><table border="1">'; //tabela pompy 4
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Pompa Nr 4: ';
            if($stan_pump_4==1){
                        echo '<a href="ustawienia_pomp.php?stan__pump_4=0"
                        > <img src="images/on.png"> </a>';  }
                        else {      echo '<a
href="ustawienia_pomp.php?stan__pump_4=1">
                         <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
```

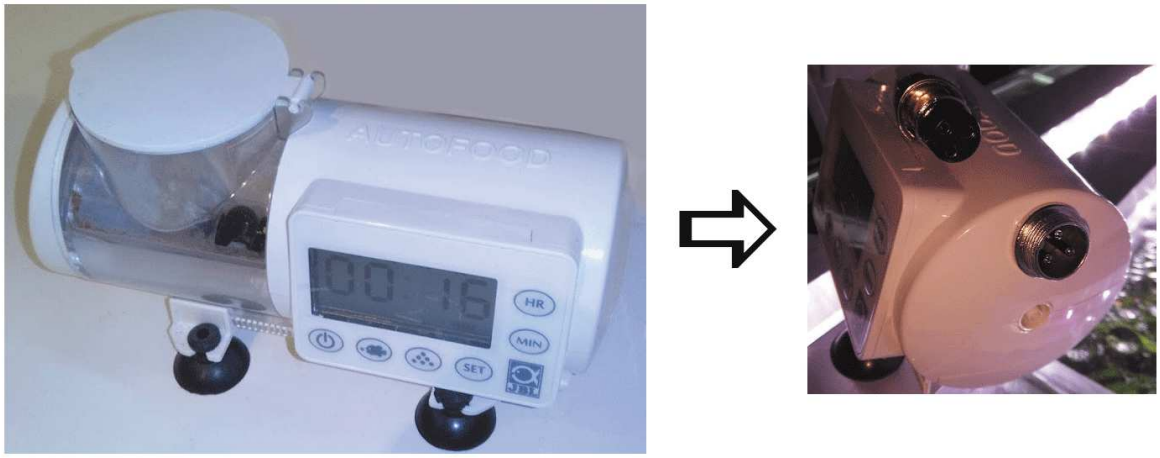
```
            <br>Pompa Nr 4 wlacza sie o godzinie: <b>' . $godzON_pump_4 .
'</b> i podaje
            jednorazowo <b>' . $Ile_dawek_pump_4 . ' ml </b>plynu <br>';

echo '</center></td></tr>
            <tr><td align="center">';
//ustawienia godziny dla pompy 4
echo ' <form method="post" name="pompa4"
            action="ustawienia_pomp.php">
            Zmien godzine pompy nr 4 (wpisz wartosc w formacie HH:MM:SS)
            <input size="5" type="text"  name="godzina_pump_4">
            <input type="submit" name="zmien_godzina_pump_4"
value="Ustaw"></form>';

$godzina_pump_4 = $_POST['godzina_pump_4'];
$zmien_godzina_pump_4 = $_POST['zmien_godzina_pump_4'];
if(isset($zmien_godzina_pump_4)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql4 = "UPDATE PUMPS SET godzON='$godzina_pump_4' WHERE Lp='4'";
if($conn->query($sql4) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:4 ' .  $conn->error;}}

//ustawienia ilosci ml dla pompy
echo ' <form method="post" name="dawka_pompa4"
            action="ustawienia_pomp.php">
            Zmien dawke dla pompy nr 4 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_pump_4">  ml
            <input type="submit" name="zmien_dawka_pump_4"
value="Ustaw"></form>';

$dawka_pump_4 = $_POST['dawka_pump_4'];
$zmien_dawka_pump_4 = $_POST['zmien_dawka_pump_4'];

if(isset($zmien_dawka_pump_4)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql4 = "UPDATE PUMPS SET Ile_dawek='$dawka_pump_4' WHERE Lp='4'";
if($conn->query($sql4) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_pomp.php");
} else {echo 'Error:4 ' .  $conn->error;}
}
echo '</td></tr></table>';
echo '</table>';//tabela glowna
echo '<br>';
//koniec pompy 4
$conn->close();
echo '<a href="korekta_pomp.php"><center>
<button>  KOREKTA DAWOKWANIA </button></a>';
echo '<a href="sterownik.php"><center>
<button> << powrot << </button></a>';
?>
```

Źródło: opracowanie własne

5.8 Automatyczne karmienie

Karmienie zwierząt również zostało zautomatyzowane. Podobnie jak w przypadku sterowania pompami dozującymi nawozy opisanymi w rozdziale 5.7, tak i w tym przypadku należało wykonać małą przeróbkę karmika automatycznego poprzez wyprowadzenie pinów w celu sterowania z zewnątrz, co zostało pokazane na rysunku 5.15.

Rys. 5.15. karmik automatyczny JBL oraz wykonana przeróbka



Źródło: zdjęcia własne (obróbka za pomocą programu Gimp)

Zasada podłączenia i działania jest analogiczna do tych opisanych w rozdziale dotyczącym pracy pomp dozujących (patrz rozdział: 5.7). W miejsce pompy podłączony jest karmik z tą różnicą, że zasilanie zmniejszone jest do 4,5 V.
Kod realizujący pracę karmidła przedstawia się następująco:

Listing 5.13: kod realizujący działanie karmidła

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      GPIO.setup(2, GPIO.OUT) #przekaznik karmika
```

```
      GPIO.output(2, 1) #przekaznik karmika
      aktywne_karm_1 = 0
      godz_karm_1 = 0
      dawka_karm_1 = 0
      znak_karm_1 = 0
      aktywne_karm_2 = 0
      godz_karm_2 = 0
      dawka_karm_2 = 0
      znak_karm_2 = 0
      aktywne_karm_3 = 0
      godz_karm_3 = 0
      dawka_karm_3 = 0
      znak_karm_3 = 0
      aktywne_karm_4 = 0
      godz_karm_4 = 0
      dawka_karm_4 = 0
      znak_karm_4 = 0


def zapytaj_db():#aby nirobic ciaglych zapytan do bazy
      #sprawdzenie ustawien karmienia
      c=[0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT godz FROM Karmik" )
      i=0
      for row in cur.fetchall():
            dane=row[0]
            c[i]=dane
            i=i+1
      global godz_karm_1
      global godz_karm_2
      global godz_karm_3
      global godz_karm_4

      godz_karm_1 = c[0]
      godz_karm_2 = c[1]
      godz_karm_3 = c[2]
      godz_karm_4 = c[3]

      #odczyt ile pokarmu ma dozowac
      c=[0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT dawka FROM Karmik" )
      i=0
      for row in cur.fetchall():
            dane=row[0]
            #print dane
            c[i]=dane
            i=i+1
      global dawka_karm_1
      global dawka_karm_2
      global dawka_karm_3
      global dawka_karm_4

      dawka_karm_1 = c[0]
      dawka_karm_2 = c[1]
      dawka_karm_3 = c[2]
      dawka_karm_4 = c[3]


      # odczyt czy aktywne
```

```
      c=[0,0,0,0]
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT aktywne FROM Karmik" )
      i=0
      for row in cur.fetchall():
            dane=row[0]
            c[i]=dane
            i=i+1
      global aktywne_karm_1
      global aktywne_karm_2
      global aktywne_karm_3
      global aktywne_karm_4

      aktywne_karm_1 = c[0]
      aktywne_karm_2 = c[1]
      aktywne_karm_3 = c[2]
      aktywne_karm_4 = c[3]


      #sprawdzenie zadanych dawek
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT czas_dawki FROM Dawki" )
      print "pytam o zadane czasy dawek ";
      i=0
      for row in cur.fetchall():
            dane=row[0]
            print dane
            a[i]=dane
            i=i+1

      global czas_dawki_pomp
      global czas_dawki_karmienia

      czas_dawki_pomp = a[0]
      czas_dawki_karmienia = a[1]

      db.close()
      return

ustawienia()
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                     user="jakobe",
                     passwd=„tajnehaslo”,
                     db="akwarium")
    #zmien wartosc w bazie i stowrz plik
      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      #cur.execute("UPDATE stany_urzadzen SET stan=1 WHERE Lp=4" )
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      #po zmianie jesli nie ma pliku to go tworz
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")

      db.close()

def karmienie():
```

```
      dev_ON = GPIO.LOW       #mod przekaznikow sterowany stanem niskim
      dev_OFF= GPIO.HIGH      #co oznacza ze przekaznik zalacza urzadzenie
przy GPIO.LOW
      now = time.strftime("%H:%M:%S") #+ "\n"
      teraz = str(now) #aby mozna bylo porownac

      if aktywne_karm_1==0:  #karmienie 1
            GPIO.output(2, dev_OFF)
      else:

            znak_karm_1 = godz_karm_1 + datetime.timedelta(0,5) #+ 5 sek
        #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp
            ON_karm_1 = str(godz_karm_1)
            znacznik_karm_1 = str(znak_karm_1)
            if ((teraz > ON_karm_1) and (teraz < znacznik_karm_1)):
                  print "karmienie 1 - karmik podaje pokarm"
                  GPIO.output(2, dev_ON)
                  dawkowanie_karmienia = dawka_karm_1 *
czas_dawki_karmienia
                  time.sleep(dawkowanie_karmienia)
                  GPIO.output(2, dev_OFF)

            else:
                  GPIO.output(2, dev_OFF)

      if aktywne_karm_2==0:  #karmienie 2
            GPIO.output(2, dev_OFF)
      else:

            znak_karm_2 = godz_karm_2 + datetime.timedelta(0,5) #+ 5 sek
        #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp
            ON_karm_2 = str(godz_karm_2)
            znacznik_karm_2 = str(znak_karm_2)
            if ((teraz > ON_karm_2) and (teraz < znacznik_karm_2)):
                  print "karmienie 2 - karmik podaje pokarm"
                  GPIO.output(2, dev_ON)
                  dawkowanie_karmienia = dawka_karm_2 *
czas_dawki_karmienia
                  time.sleep(dawkowanie_karmienia)
                  GPIO.output(2, dev_OFF)

            else:
                  GPIO.output(2, dev_OFF)

      if aktywne_karm_3==0:  #karmienie 3
            GPIO.output(2, dev_OFF)
      else:

            znak_karm_3 = godz_karm_3 + datetime.timedelta(0,5) #+ 5 sek
        #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp
            ON_karm_3 = str(godz_karm_3)
            znacznik_karm_3 = str(znak_karm_3)
            if ((teraz > ON_karm_3) and (teraz < znacznik_karm_3)):
                  print "karmienie 3 - karmik podaje pokarm"
                  GPIO.output(2, dev_ON)
                  dawkowanie_karmienia = dawka_karm_3 *
czas_dawki_karmienia
                  time.sleep(dawkowanie_karmienia)
                  GPIO.output(2, dev_OFF)

            else:
```

```
                  GPIO.output(2, dev_OFF)


      if aktywne_karm_4==0:  #karmienie 4
            GPIO.output(2, dev_OFF)
      else:

            znak_karm_4 = godz_karm_4 + datetime.timedelta(0,5) #+ 5 sek
         #tyle +/- trwa cykl skryptu zeby nie bawic sie z timestamp
            ON_karm_4 = str(godz_karm_4)
            znacznik_karm_4 = str(znak_karm_4)
            if ((teraz > ON_karm_4) and (teraz < znacznik_karm_4)):
                  print "karmienie 4 - karmik podaje pokarm"
                  GPIO.output(2, dev_ON)
                  dawkowanie_karmienia = dawka_karm_4 *
czas_dawki_karmienia
                  time.sleep(dawkowanie_karmienia)
                  GPIO.output(2, dev_OFF)

            else:
                  GPIO.output(2, dev_OFF)

while True:
      global delay
      #os.system("clear")

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
            karmienie()
```

Źródło: opracowanie własne

Możliwość zmiany ustawień karmika realizowane jest z poziomu przeglądarki internetowej. Odpowiedni kod w PHP modyfikuje zawartość bazy:

Listing 5.14: kod modyfikujący ustawienia automatycznego karmienia

```
<?php
$plik = file('private/accessdb.txt');
      //zczytanie pliku (do tablicy po linii)
$servername = chop($plik[0]); //chop() usuwa znaki konca linii
$username = chop($plik[1]);
$password = chop($plik[2]);
$dbname = chop($plik[3]);

echo '<center><h1>USTAWIENIA KARMIENIA:</h1>
<br>(Jedna dawka karmienia to podawanie pokarmu przez  0,5 sekundy)';
//odczytanie ktore karmienia sa aktywne
```

```
$conn = new mysqli($servername, $username, $password, $dbname);
 $sql = "SELECT aktywne FROM Karmik";
 $result = $conn->query($sql);
 for ($aktywne=array(); $row=$result->fetch_assoc();
$aktywne[]=$row[aktywne]);

$aktywne_karmienie_1 = $aktywne[0];
$aktywne_karmienie_2 = $aktywne[1];
$aktywne_karmienie_3 = $aktywne[2];
$aktywne_karmienie_4 = $aktywne[3];

//odczytanie godzin karmienia
 $sql = "SELECT godz FROM Karmik";
 $result = $conn->query($sql);
 for ($godz=array(); $row=$result->fetch_assoc(); $godz[]=$row[godz]);

 $godz_karm_1 = $godz[0];
 $godz_karm_2 = $godz[1];
 $godz_karm_3 = $godz[2];
 $godz_karm_4 = $godz[3];


//odczytanie dawek poszczegolnego karmienia
 $sql = "SELECT dawka FROM Karmik";
 $result = $conn->query($sql);
 for ($dawka=array(); $row=$result->fetch_assoc(); $dawka[]=$row[dawka]);

 $dawka_karm_1 = $dawka[0];
 $dawka_karm_2 = $dawka[1];
 $dawka_karm_3 = $dawka[2];
 $dawka_karm_4 = $dawka[3];

//zmiana ustawien pomp
$stan__karm_1=$_GET['stan__karm_1'];
$stan__karm_2=$_GET['stan__karm_2'];
$stan__karm_3=$_GET['stan__karm_3'];
$stan__karm_4=$_GET['stan__karm_4'];


if(isset($_GET['stan__karm_1'])){
      $zmien_stan__karm_1 = "UPDATE Karmik SET aktywne='" . $stan__karm_1 .
"' WHERE Lp='1'";
            if($conn->query($zmien_stan__karm_1) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_karmienia.php");
                  } else {echo 'Error' .  $conn->error;}
}if(isset($_GET['stan__karm_2'])){
      $zmien_stan__karm_2 = "UPDATE Karmik SET aktywne='" . $stan__karm_2 .
"' WHERE Lp='2'";
            if($conn->query($zmien_stan__karm_2) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_karmienia.php");
                  } else {echo 'Error' .  $conn->error;}
}if(isset($_GET['stan__karm_3'])){
      $zmien_stan__karm_3 = "UPDATE Karmik SET aktywne='" . $stan__karm_3 .
"' WHERE Lp='3'";
            if($conn->query($zmien_stan__karm_3) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_karmienia.php");
                  } else {echo 'Error' .  $conn->error;}
}if(isset($_GET['stan__karm_4'])){
```

```
      $zmien_stan__karm_4 = "UPDATE Karmik SET aktywne='" . $stan__karm_4 .
"' WHERE Lp='4'";
            if($conn->query($zmien_stan__karm_4) === TRUE) {
                  $zmiana = fopen("change.txt", "w");
                  header("Location: ustawienia_karmienia.php");
                  } else {echo 'Error' .  $conn->error;}
}


echo '<center><table border="1">'; //tabela karmienia 1
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Karmienie Nr 1: ';
            if($aktywne_karmienie_1==1){
                        echo '<a
href="ustawienia_karmienia.php?stan__karm_1=0">
                         <img src="images/on.png"> </a>';   }
                        else {      echo '<a
href="ustawienia_karmienia.php?stan__karm_1=1">
                        <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
            <br>Karmienie 1 o godzinie: <b>' . $godz_karm_1 . '</b> i
podaje
            jednorazowo <b>' . $dawka_karm_1 . ' dawek </b> pokarmu <br>';

echo '</center></td></tr>
            <tr><td align="center">';


//ustawienia godziny dla karmienia 1
echo ' <form method="post" name="karmienie1"
            action="ustawienia_karmienia.php">
            Zmien godzine karmienia nr 1 (wpisz wartosc w formacie
HH:MM:SS)
            <input size="5" type="text"  name="godzina_karm_1">
            <input type="submit" name="zmien_godzina_karm_1"
value="Ustaw"></form>';

$godzina_karm_1 = $_POST['godzina_karm_1'];
$zmien_godzina_karm_1 = $_POST['zmien_godzina_karm_1'];
if(isset($zmien_godzina_karm_1)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql1 = "UPDATE Karmik SET godz='$godzina_karm_1' WHERE Lp='1'";
if($conn->query($sql1) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:2 ' .  $conn->error;}}

//ustawienia dawki dla karmienia 1
echo ' <form method="post" name="dawka_karmienie1"
            action="ustawienia_karmienia.php">
            Zmien dawke dla karmienia nr 1 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_karm_1">
            <input type="submit" name="zmien_dawka_karm_1"
value="Ustaw"></form>';

$dawka_karm_1 = $_POST['dawka_karm_1'];
$zmien_dawka_karm_1 = $_POST['zmien_dawka_karm_1'];

if(isset($zmien_dawka_karm_1)){
```

```
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql1 = "UPDATE Karmik SET dawka='$dawka_karm_1' WHERE Lp='1'";
if($conn->query($sql1) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:1 ' .  $conn->error;}

}
echo '</td></tr></table>';
echo '</table>';
echo '<br>';
//koniec kamienia  1



echo '<center><table border="1">'; //tabela karmienia 2
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Karmienie Nr 2: ';
            if($aktywne_karmienie_2==1){
                        echo '<a
href="ustawienia_karmienia.php?stan__karm_2=0">
                         <img src="images/on.png"> </a>';   }
                        else {      echo '<a
href="ustawienia_karmienia.php?stan__karm_2=1">
                        <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
            <br>Karmienie 2 o godzinie: <b>' . $godz_karm_2 . '</b> i
podaje
            jednorazowo <b>' . $dawka_karm_2 . ' dawek </b> pokarmu <br>';

echo '</center></td></tr>
            <tr><td align="center">';


//ustawienia godziny dla karmienia 2
echo ' <form method="post" name="karmienie2"
            action="ustawienia_karmienia.php">
            Zmien godzine karmienia nr 2 (wpisz wartosc w formacie
HH:MM:SS)
            <input size="5" type="text"  name="godzina_karm_2">
            <input type="submit" name="zmien_godzina_karm_2"
value="Ustaw"></form>';

$godzina_karm_2 = $_POST['godzina_karm_2'];
$zmien_godzina_karm_2 = $_POST['zmien_godzina_karm_2'];
if(isset($zmien_godzina_karm_2)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql2 = "UPDATE Karmik SET godz='$godzina_karm_2' WHERE Lp='2'";
if($conn->query($sql2) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:2 ' .  $conn->error;}}

//ustawienia dawki dla karmienia 2
echo ' <form method="post" name="dawka_karmienie2"
            action="ustawienia_karmienia.php">
```

```
            Zmien dawke dla karmienia nr 2 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_karm_2">
            <input type="submit" name="zmien_dawka_karm_2"
value="Ustaw"></form>';

$dawka_karm_2 = $_POST['dawka_karm_2'];
$zmien_dawka_karm_2 = $_POST['zmien_dawka_karm_2'];

if(isset($zmien_dawka_karm_2)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql2 = "UPDATE Karmik SET dawka='$dawka_karm_2' WHERE Lp='2'";
if($conn->query($sql2) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:2 ' .  $conn->error;}

}
echo '</td></tr></table>';
echo '</table>';
echo '<br>';
//koniec kamienia  2


echo '<center><table border="1">'; //tabela karmienia 3
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Karmienie Nr 3: ';
            if($aktywne_karmienie_3==1){
                        echo '<a
href="ustawienia_karmienia.php?stan__karm_3=0">
                         <img src="images/on.png"> </a>';   }
                        else {      echo '<a
href="ustawienia_karmienia.php?stan__karm_3=1">
                         <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
            <br>Karmienie 3 o godzinie: <b>' . $godz_karm_3 . '</b> i
podaje
            jednorazowo <b>' . $dawka_karm_3 . ' dawek </b> pokarmu <br>';

echo '</center></td></tr>
            <tr><td align="center">';


//ustawienia godziny dla karmienia 3
echo ' <form method="post" name="karmienie3"
            action="ustawienia_karmienia.php">
            Zmien godzine karmienia nr 3 (wpisz wartosc w formacie
HH:MM:SS)
            <input size="5" type="text"  name="godzina_karm_3">
            <input type="submit" name="zmien_godzina_karm_3"
value="Ustaw"></form>';

$godzina_karm_3 = $_POST['godzina_karm_3'];
$zmien_godzina_karm_3 = $_POST['zmien_godzina_karm_3'];
if(isset($zmien_godzina_karm_3)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql3 = "UPDATE Karmik SET godz='$godzina_karm_3' WHERE Lp='3'";
```

```
if($conn->query($sql3) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:3 ' .  $conn->error;}}

//ustawienia dawki dla karmienia 3
echo ' <form method="post" name="dawka_karmienie3"
            action="ustawienia_karmienia.php">
            Zmien dawke dla karmienia nr 3 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_karm_3">
            <input type="submit" name="zmien_dawka_karm_3"
value="Ustaw"></form>';

$dawka_karm_3 = $_POST['dawka_karm_3'];
$zmien_dawka_karm_3 = $_POST['zmien_dawka_karm_3'];

if(isset($zmien_dawka_karm_3)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql3 = "UPDATE Karmik SET dawka='$dawka_karm_3' WHERE Lp='3'";
if($conn->query($sql3) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:3 ' .  $conn->error;}

}
echo '</td></tr></table>';
echo '</table>';
echo '<br>';
//koniec kamienia  3


echo '<center><table border="1">'; //tabela karmienia 4
echo '<tr><td>';
echo'<table border="0">';
echo '<tr><td><center>Karmienie Nr 4: ';
            if($aktywne_karmienie_4==1){
                        echo '<a
href="ustawienia_karmienia.php?stan__karm_4=0">
                        <img src="images/on.png"> </a>';    }
                        else {      echo '<a
href="ustawienia_karmienia.php?stan__karm_4=1">
                        <img src="images/off.png"> </a>';}
            echo '</center></td></tr><tr><td><center>
            <br>Karmienie 4 o godzinie: <b>' . $godz_karm_4 . '</b> i
podaje
            jednorazowo <b>' . $dawka_karm_4 . ' dawek </b> pokarmu <br>';

echo '</center></td></tr>
            <tr><td align="center">';
//ustawienia godziny dla karmienia 4
echo ' <form method="post" name="karmienie4"
            action="ustawienia_karmienia.php">
            Zmien godzine karmienia nr 4 (wpisz wartosc w formacie
HH:MM:SS)
            <input size="5" type="text"  name="godzina_karm_4">
            <input type="submit" name="zmien_godzina_karm_4"
value="Ustaw"></form>';
```

```
$godzina_karm_4 = $_POST['godzina_karm_4'];
$zmien_godzina_karm_4 = $_POST['zmien_godzina_karm_4'];
if(isset($zmien_godzina_karm_4)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);}
$sql4 = "UPDATE Karmik SET godz='$godzina_karm_4' WHERE Lp='4'";
if($conn->query($sql4) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:4 ' .  $conn->error;}}

//ustawienia dawki dla karmienia 4
echo ' <form method="post" name="dawka_karmienie4"
            action="ustawienia_karmienia.php">
            Zmien dawke dla karmienia nr 4 (wpisz wartosc)
            <input size="5" type="text"  name="dawka_karm_4">
            <input type="submit" name="zmien_dawka_karm_4"
value="Ustaw"></form>';

$dawka_karm_4 = $_POST['dawka_karm_4'];
$zmien_dawka_karm_4 = $_POST['zmien_dawka_karm_4'];

if(isset($zmien_dawka_karm_4)){
if($conn ->connect_error) {
      die('Nie moge sie polaczyc: ' . $conn->connect_error);
}

$sql4 = "UPDATE Karmik SET dawka='$dawka_karm_4' WHERE Lp='4'";
if($conn->query($sql4) === TRUE) {
      $zmiana = fopen("change.txt", "w");
      header("Location: ustawienia_karmienia.php");
} else {echo 'Error:4 ' .  $conn->error;}

}
echo '</td></tr></table>';
echo '</table>';
echo '<br>';
//koniec kamienia  4

$conn->close();
//echo '<a href="korekta_pomp.php"><center>
<button>  KOREKTA DAWOKWANIA </button></a>';

echo '<a href="sterownik.php"><center>
<button> << powrot << </button></a>';
?>
```

Źródło: opracowanie własne

5.9 Sterowanie napowietrzaczem – serwomechanizm

Jednym z podstawowych parametrów, jakie należy utrzymywać na odpowiednim poziomie, jest poziom natlenienia wody. Do natlenienia w akwarystyce wykorzystywany jest m.in. napowietrzacz, który posiada regulację mechaniczną z wykorzystaniem potencjometru. Aby można było nim sterować z poziomu komputera, trzeba wykorzystać serwomechanizm. W projekcie użyto serwa modelarskiego jak na rysunku 5.16.

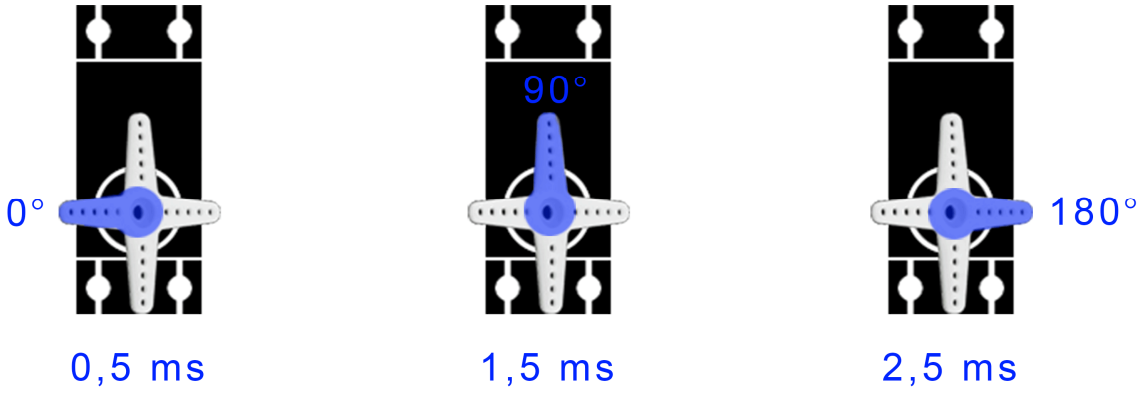
Rys. 5.16. serwo modelarskie

Źródło: zdjęcie własne

Sterowanie serwem odbywa się poprzez generowanie odpowiedniego sygnału PWM - Pulase Width Modulation - czyli modulacji szerokości impulsu. Technika ta stosowana jest do kodowania komunikatu w sygnał pulsujący w taki sposób, aby umożliwić sterowanie mocą dostarczaną do urządzeń elektrycznych, zwłaszcza do bezwładnościowych obciążeń, takich jak silniki czy serwa[8]. Serwa pracują w częstotliwości 50 Hz czyli w cyklach trwających 20 ms [9]. W zależności od czasu wypełnienia całego cyklu ramię serwa odpowiednio się wychyla (zakres czasu impulsu, w jakim pracuje serwo to 0.5 do 2.5 ms). Kąty wychylenia ramienia serwa w zależności od czasu impulsu przedstawia rysunek 5.17.

[8] https://pl.wikipedia.org/wiki/Modulacja_szerokości_impulsów (data odczytu: 27-11-2017 r.)
[9] Merta A.: PWM dla serw *Miesięcznik "Młody Technik"*, sierpień 2015 str. 87

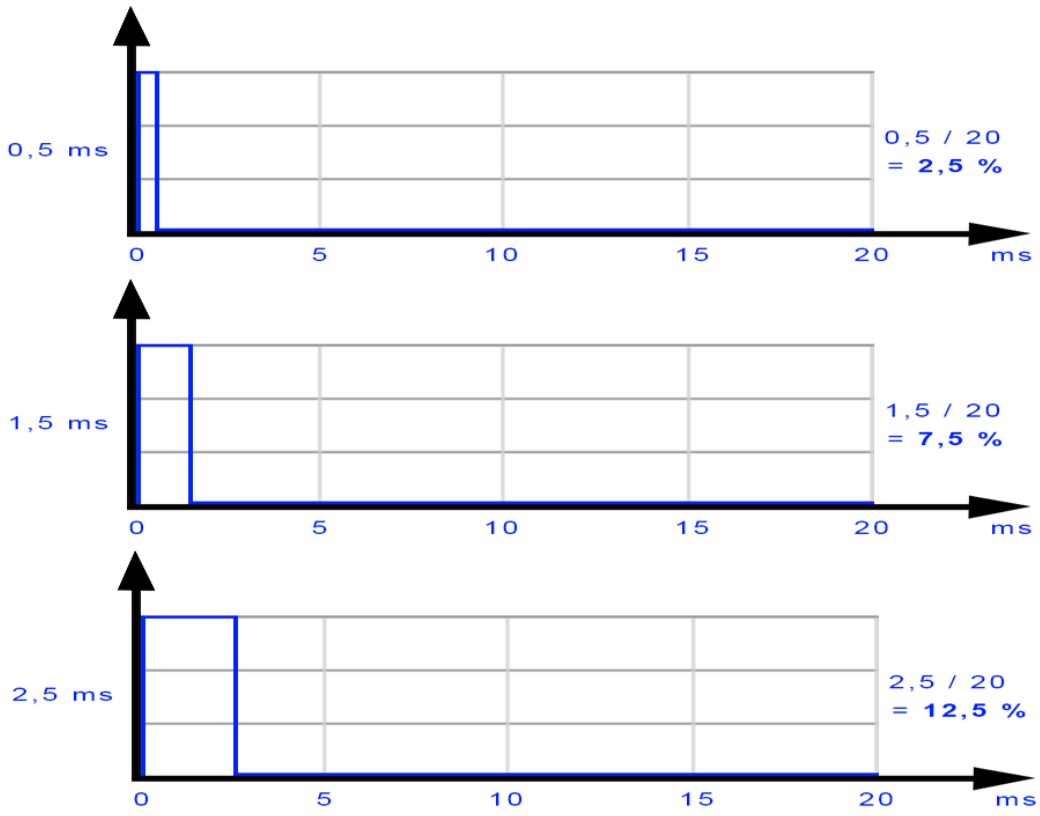
Rys. 5.17. ułożenie orczyka w zależności od czasu impulsu



Źródło: opracowanie własne

Czasy podawane są w procentach wypełnienia cyklu (20 ms = 100 %) zgodnie z diagramem przedstawionym na rysunku 5.18.

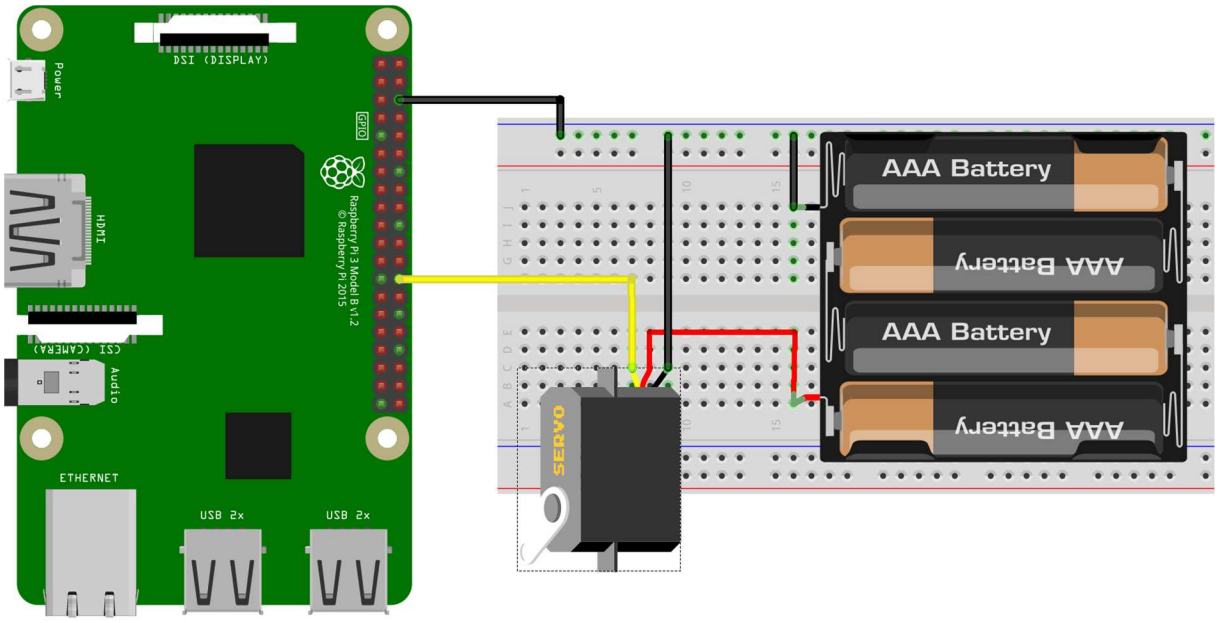
Rys. 5.18. czasy wypełnienia pojedynczego sygnału pracy serwa



Źródło: opracowanie własne

Podłączenie serwa do Rasberry Pi przedstawia schemat 5.19. Aby serwo pracowało poprawnie, należy zastosować zasilanie zewnętrzne (4,8-6,0 V), ze względu na to, że Raspberry Pi na wyjściu GPIO podaje napięcie o zbyt niskim amperażu.

Rys. 5.19. schemat podłączenia serwa do Raspberry Pi wraz z zewnętrznym zasilaniem



Źródło: opracowanie własne (za pomocą programu fritzing)

Kod obsługujący pracę serwa przedstawia się następująco:

Listing 5.15: kod obsługujący pracę serwa

```
# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import os, sys, time
import os.path
import MySQLdb
import datetime
from time import strftime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

def ustawienia():
      GPIO.setup(7, GPIO.OUT) #servo
      ust_servo = 0

def zapytaj_db():#aby nirobic ciaglych zapytan do bazy
      #sprawdzenie zadanego polozenia serva
      cur = db.cursor() #create a Cursor object.
      cur.execute("SELECT wartosc FROM servo" )
      print "pytam o ust serva ";
```

```
      for row in cur.fetchall():
            servo=row[0]
            print servo

      global ust_servo
      ust_servo=servo

      db.close()
      return

ustawienia()
zapytaj_db()

def update_stanu_w_bazie(stan_urzadzenia, Lp):
      db = MySQLdb.connect(host="localhost",
                  user="jakobe",
                  passwd=„tajnehaslo”,
                  db="akwarium")
    #zmien wartosc w bazie i stowrz plik
      print 'robie update do bazy'
      cur = db.cursor() #create a Cursor object.
      #cur.execute("UPDATE stany_urzadzen SET stan=1 WHERE Lp=4" )
      cur.execute("UPDATE stany_urzadzen SET stan=%s WHERE Lp=%s" %
(stan_urzadzenia, Lp))
      db.commit()
      #po zmianie jesli nie ma pliku to go tworz
      if not os.path.exists("change.txt"):
            os.mknod("change.txt")

      db.close()

def ustawienie_serva():
      print "servo ustawione jest na: "
      print ust_servo
      p = GPIO.PWM(7,50)
      p.start(7.5)
      p.ChangeDutyCycle(ust_servo)
      time.sleep(1)
      return


while True:
      global delay
      #os.system("clear")

      if os.path.exists("change.txt"):
            print "plik istnieje"
            zapytaj_db()
            ustawienie_serva() #aby servo nie chodzilo non stop
            #ustawia sie po zmianie w bazie inaczej przelczenie
            #przekaznika powoduje mikroruchy
            os.remove("change.txt")
            print "usuwam plik"

      else:
            print "nie ma pliku"
```

Źródło: opracowanie własne

Ustawienie pozycji orczyka dokonywane jest poprzez wskazanie, z jaka mocą ma działać napowietrzacz. W zależności od wartości (od 2,5 do 12,5) zapisanej w bazie danych orczyk serwa steruje potencjometrem, który zwiększa lub zmniejsza ilość tlenu dostarczanego do zbiornika.

Listing 5.16: skrypt PHP pozwalający sterować serwomechanizmem

```
<?php
header('refresh: 10;');
$plik = file('private/accessdb.txt');
      //zczytanie pliku (do tablicy po linii)
$servername = chop($plik[0]); //chop() usuwa znaki konca linii
$username = chop($plik[1]);
$password = chop($plik[2]);
$dbname = chop($plik[3]);

//zmiana ustawien serva
$plozenie_serwa=$_GET['plozenie_serwa'];
if(isset($_GET['plozenie_serwa'])){
      $zmien_plozenie_serwa = "UPDATE servo SET wartosc='" .
$plozenie_serwa . "' WHERE Lp='1'";
      if($conn->query($zmien_plozenie_serwa) === TRUE) {
            $zmiana = fopen("change.txt", "w");
            header("Location: sterownik.php");
            } else {echo 'Error: ' .  $conn->error;}
}


echo '<center>
      <table border="0">
      <tr><td colspan="8">
      <center><h1>Sterownik akwariowy</h1></center></td></tr>
      echo '<td width="80" align="center"><font size="1">moc:</font>';

       $spr_servo = "SELECT wartosc FROM servo WHERE Lp='1'";
            $result = $conn->query($spr_servo);
            if($result->num_rows > 0){
                  while($wiersz = $result->fetch_assoc()) {
                  $servo = $wiersz["wartosc"];
                  }
            }
            else {
                  echo 'BLAD';
            }

            $j = $servo/1.25-0.01;
            echo '<table border="0" cellspacing="0" cellpadding="0"><tr>';
            for ($i=1; $i<10; $i++) {
                  $k = 1.25+$i*1.25; //wartosci do insertu
                  if($i<=$j&&$stan_230V_1!=0){
                  echo '<td align="center" valign="bottom">
                  <a href="sterownik.php?plozenie_serwa=' . $k . '">
                  <img src="images/redbar.png" height="' . 3*$i .'"
width="5">
                  </a></td>';
                  } else echo '<td align="center" valign="bottom">
                  <a href="sterownik.php?plozenie_serwa=' . $k . '">
```

```
                  <img src="images/greybar.png" height="' . 3*$i .'"
width="5">
                  </a></td>';
            }
            echo '</tr><tr>';
            for ($i=1; $i<10; $i++) {
                  echo'<td align="center" valign="bottom"><font size="2">'
. $i . '</td>';
                  }
            echo '</tr></table>';

      echo '</td></tr>';
      </tr>

      </table>';


?>
```

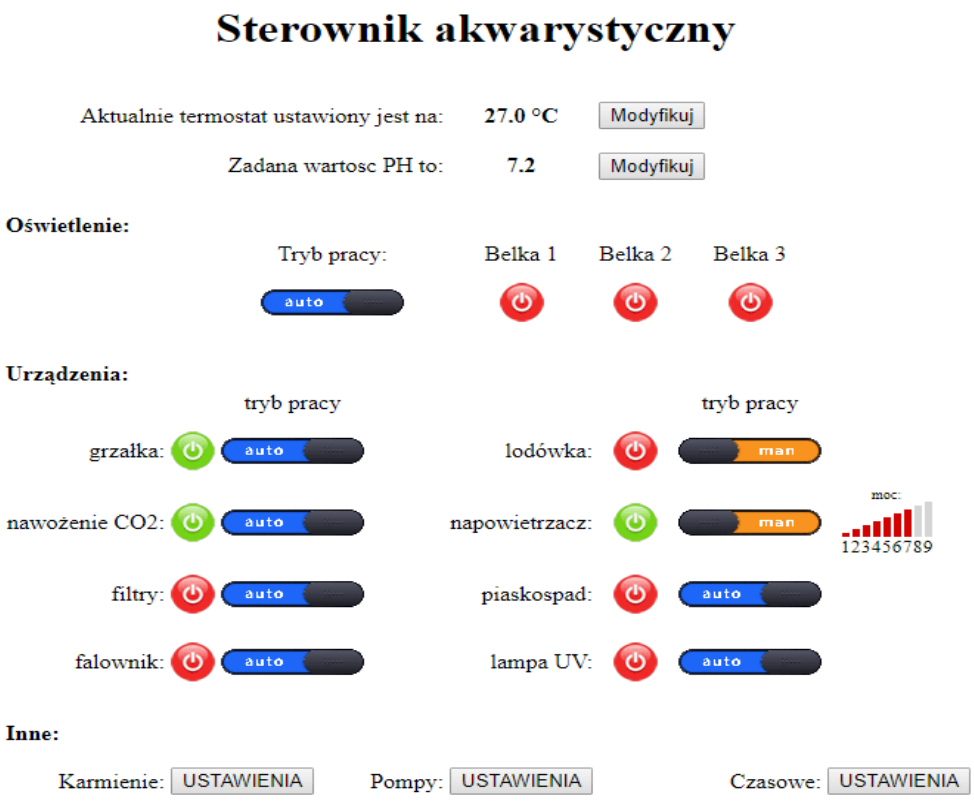
Źródło: opracowanie własne

6 Obsługa - baza danych i WWW

Przy małych projektach z użyciem pojedynczych czujników i pojedynczych urządzeń sterujących można stosować np. zapis stanów poszczególnych czujników do pliku. Przy realizacji większego projektu zasadne jest wykorzystanie do tego celu bazy danych, w której oprócz stanów logicznych poszczególnych czujników zapisywane są różne dane dotyczące działania całego sterownika.

Obsługa sterownika będzie realizowana poprzez stronę www, dlatego wykorzystano PHP[10], za pomocą którego zaimplementowano interface użytkownika (rysunek 6.1 przedstawia zrzut ekranu interface'u), oraz bazę danych (w tym przypadku MariaDB[11]).

Rys. 6.1. interface użytkownika pozwalający na zarządzanie sterownikiem



Źródło: opracowanie własne

[10] PHP jest językiem, który w czasie rzeczywistym generuje strony internetowe, poprzez wykonywanie skryptów po stronie serwera Źródło: https://pl.wikipedia.org/wiki/PHP (data odczytu: 12-03-2018 r.)

[11] MariaDB jest bazą na licencji GPL i jest alternatywą dla popularnego MySQLa. Źródło: https://pl.wikipedia.org/wiki/MariaDB (data odczytu: 24-02-2018 r.)

Poniżej znajduje się zrzut bazy danych wykonany za pomocą PhpMyAdmin[12]:

Listing 6.1: baza danych "akwarium"

```
-- phpMyAdmin SQL Dump
-- version 4.6.6deb4
-- https://www.phpmyadmin.net/
--
-- Host: localhost:3306
-- Czas generowania: 08 Mar 2018, 17:42
-- Wersja serwera: 10.1.23-MariaDB-9+deb9u1
-- Wersja PHP: 5.6.33-0+deb8u1

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Baza danych: `akwarium`
--

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `czasowe`
--

CREATE TABLE `czasowe` (
  `Lp` int(1) NOT NULL,
  `urzadzenie` varchar(15) COLLATE utf8_polish_ci NOT NULL,
  `godz_on` time NOT NULL,
  `godz_off` time NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `czasowe`
--

INSERT INTO `czasowe` (`Lp`, `urzadzenie`, `godz_on`, `godz_off`) VALUES
(1, 'filtry', '08:45:00', '19:00:00'),
(2, 'lampa_UV', '11:15:00', '18:30:00'),
(3, 'falownik', '07:15:00', '19:50:00'),
(4, 'piaskospad', '20:55:39', '21:05:39');

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `Dawki`
--

CREATE TABLE `Dawki` (
```

[12] PhpMyAdmin jest narzędziem wspomagającym zarządzanie bazą danych. Źródło: https://pl.wikipedia.org/wiki/PhpMyAdmin (data odczytu: 24-02-2018 r.)

```
  `Lp` int(1) NOT NULL,
  `urzadzenie` varchar(9) COLLATE utf8_polish_ci NOT NULL,
  `czas_dawki` decimal(3,3) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `Dawki`
--

INSERT INTO `Dawki` (`Lp`, `urzadzenie`, `czas_dawki`) VALUES
(1, 'pumps', 0.756),
(2, 'karmik', 0.600);

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `Karmik`
--

CREATE TABLE `Karmik` (
  `Lp` int(1) NOT NULL,
  `nr_karmienia` int(1) NOT NULL,
  `aktywne` int(11) NOT NULL,
  `godz` time NOT NULL,
  `dawka` int(2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `Karmik`
--

INSERT INTO `Karmik` (`Lp`, `nr_karmienia`, `aktywne`, `godz`, `dawka`)
VALUES
(1, 1, 1, '14:37:00', 7),
(2, 2, 1, '14:30:00', 8),
(3, 3, 0, '14:26:00', 9),
(4, 4, 1, '14:32:00', 10);

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `PH`
--

CREATE TABLE `PH` (
  `Lp` int(1) NOT NULL,
  `wartosc` decimal(5,1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `PH`
--

INSERT INTO `PH` (`Lp`, `wartosc`) VALUES
(1, 7.2);

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `PUMPS`
--
```

```
CREATE TABLE `PUMPS` (
  `Lp` int(1) NOT NULL,
  `urzadzenie` varchar(9) COLLATE utf8_polish_ci NOT NULL,
  `on_off` int(1) NOT NULL,
  `godzON` time NOT NULL,
  `Ile_dawek` int(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `PUMPS`
--

INSERT INTO `PUMPS` (`Lp`, `urzadzenie`, `on_off`, `godzON`, `Ile_dawek`)
VALUES
(1, 'pump_1', 1, '14:40:00', 10),
(2, 'pump_2', 1, '14:39:00', 4),
(3, 'pump_3', 0, '15:00:00', 15),
(4, 'pump_4', 1, '16:00:00', 16);

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `servo`
--

CREATE TABLE `servo` (
  `Lp` int(1) NOT NULL,
  `wartosc` decimal(5,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `servo`
--

INSERT INTO `servo` (`Lp`, `wartosc`) VALUES
(1, 10.00);

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `stany_urzadzen`
--

CREATE TABLE `stany_urzadzen` (
  `Lp` int(2) NOT NULL,
  `urzadzenie` varchar(9) COLLATE utf8_polish_ci NOT NULL,
  `stan` int(1) NOT NULL,
  `man_auto` int(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `stany_urzadzen`
--

INSERT INTO `stany_urzadzen` (`Lp`, `urzadzenie`, `stan`, `man_auto`)
VALUES
(1, 'LED_1', 0, 1),
(2, 'LED_2', 0, 0),
(3, 'LED_3', 0, 0),
(4, 'grzalka', 1, 1),
```

```
(5, 'lodowka', 0, 1),
(6, '230V_1', 0, 1),
(7, '230V_2', 1, 1),
(8, '230V_3', 0, 1),
(9, '230V_4', 0, 1),
(10, '230V_5', 0, 1),
(11, '230V_6', 0, 1);

-- --------------------------------------------------------

--
-- Struktura tabeli dla tabeli `Temperatura`
--

CREATE TABLE `Temperatura` (
  `Lp` int(1) NOT NULL,
  `wartosc` decimal(5,1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;

--
-- Zrzut danych tabeli `Temperatura`
--

INSERT INTO `Temperatura` (`Lp`, `wartosc`) VALUES
(1, 27.0);

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Źródło: opracowanie własne

7 Podsumowanie

Dzięki zastosowaniu inteligentnych instalacji budynkowych, przy użyciu niewielkich nakładów finansowych można zautomatyzować działanie różnych urządzeń.

Wykonany przez autora sterownik daje wiele możliwości, pozwala ograniczyć koszty i czas. Dzięki niemu można stale monitorować parametry techniczne akwarium oraz dbać o optymalne warunki fauny i flory zbiornika. Sterownik daje wiele możliwości rozbudowy na przykład o kamerę, która w czasie rzeczywistym daje możliwość stałego nadzoru nad komponentami technicznymi jak i wizualnymi akwarium (spoza miejsca, w którym akwarium się znajduje), czy o wagę, którą można wykorzystać do ważenia butli z dwutlenkiem węgla, co pozwoli na monitorowanie ilości gazu. Dostępne opcje ogranicza jedynie wyobraźnia.

Sterownik dzięki odpowiedniemu zaprogramowaniu zapewnia zwierzętom stały, optymalny dostęp do pokarmu. Dba o właściwy poziom CO_2, a także zapewnia ciekawe i estetyczne wrażenia użytkownikowi.

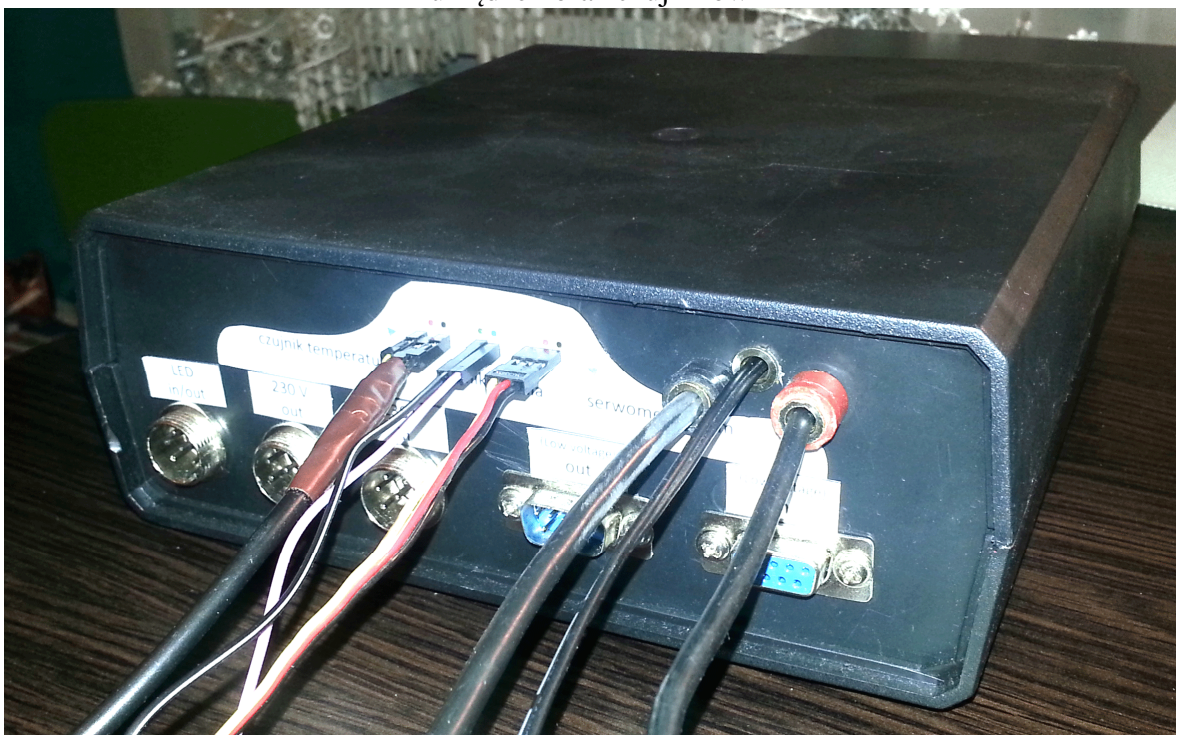
Automatyzacja wyposażenia w budynkach mieszkalnych, biurowych, czy fabrykach staje się standardem. Z powodzeniem można ją wykorzystać na każdym polu. W tym przypadku instalację zaadaptowano do zbiornika wodnego, jednak dzięki wykorzystaniu platofrm Rapberry Pi oraz Arduino można ją wykorzystać również w innych dziedzinach.

Poniżej przedstawiono fotografie wykonanego sterownika do obsługi zbiornika wodnego.

Rys. 7.1. fotografia przedstawiająca sterownik od strony przedniej, gdzie znajduje się czujnik ruchu oraz natężenia światła i dioda alarmowa sygnalizująca zalanie

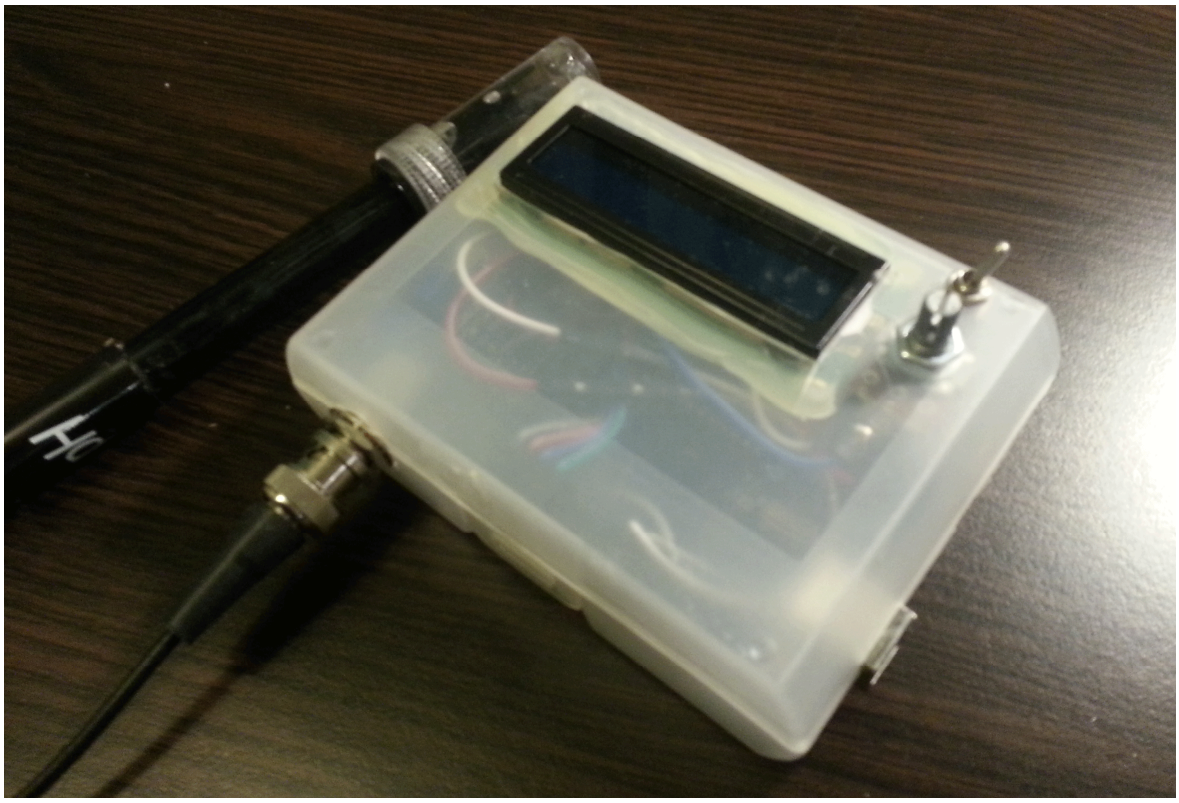
Źródło: zdjęcie własne

Rys. 7.2. fotografia przedstawiająca sterownik od strony pozwalającej na podłączenia urządzeń oraz czujników



Źródło: zdjęcie własne

Rys. 7.3 fotografia przedstawiająca moduł sterownika mierzący poziom PH



Źródło: zdjęcie własne

Rys. 7.4. sterownik podłączony do akwarium

Źródło: zdjęcie własne

Bibliografia

[1] Asadi A.: *Practical Raspnerry Pi Project* Imagine Publishing Ltd., Richmond Wielka Brytania 2015.

[2] Bolanowski B., Borkowski P.: *Budowa Systemów i urządzeń zasilających* Wydawnictwo Wyższej Szkoły Humanistyczno-Ekonomicznej w Łodzi, Łódź 2007.

[3] Dechnik M., Furtak M.: Inteligentne Budynki Dziś i Jutro *Czasopismo "Builder"*, czerwiec 2017 str. 64-67

[4] Kwaśniewski J.: *Inteligentny dom i inne systemy sterowania w 100 przykładach* Wydawnictwo BTC, Legionowo 2011.

[5] Merta A.: PWM dla serw *Miesięcznik "Młody Technik"*, sierpień 2015 str. 86-93

[6] Monk S.: *Arduino + Android for the Evil Genius*, w tłumaczeniu Watrak A.: *Arduino i Android. Niesamowite Projekty*, HELION, Gliwice 2014.

[7] Monk S.: *Programming the Raspberry Pi: Getting Started with Python*, w tłumaczeniu Janusz J.: *Raspberry Pi. Przewodnik dla programistów Pythona*, HELION, Gliwice 2014.

[8] Riley M.: *Programing Your Home: Automate with Arduino, Android, and Your Computer*, w tłumaczeniu Szczepaniak M.: *Inteligentny dom - Automatyzacja mieszkania za pomocą platformy Arduino, systemu Android i zwykłego komputera*, HELION, Gliwice 2013.

[9] Schwartz S.: *Arduino Home Automation Projects*, w tłumaczeniu Waśko Z.: *Adruino Automatyka domowa dla każdego*, HELION, Gliwice 2015.

[10] Sroczan E.: *Nowoczesne wyposażenie techniczne domu jednorodzinnego. Instalacje elektryczne* PWRIL Państwowe Wydawnictwo Rolnicze i Leśne, Poznań 2004.

[11] Upton E., Halfacree G.: *Raspberry Pi User Guide*, w tłumaczeniu Szczepaniak M.:*Raspberry Pi - Przewodnik Użytkownika*, HELION, Gliwice 2013.

Spis rysunków

Spis tabel