Dozator de bobite pentru creaturile necuvantatoare

Introducere

Proiectul va avea un senzor de temperatura, care depisteaza prezenta unei pisici infometate. Daca pisica e in apropiere atunci bobitele vor cadea, iar daca cumva pisica doreste sa manance si mai mult, acest dozator are un timer, care tine cont ca in urmatoarele 6 ore nu va mai lasa bobitele sa cada. Atunci cand este permisa caderea bobitelor, un led va semnala asta.

Descriere generală

Schema bloc: https://imgur.com/a/6gjnXB1

https://imgur.com/a/9QGRXnq

Hardware Design

Folosesc:

- Arduino uno
- board
- senzor de temperatura
- fire
- servo

Software Design

Folosesc:

- Arduino IDE
- <Servo.h>

Jurnal

Pe partea de hardware, am testat componentele si le am atasat rezultatul fiind: https://imgur.com/NDp6wOu

Pe partea de software, am modificat ca rezultatul sa fie folosibil si pentru oameni (avand temperatura mai mica) si timer-ul de 10 secunde ceea ce mi a permis sa testez complet functionalitatile: https://imgur.com/5bBw9a7

Dar nu va faceti griji, l-am testat si pe motan: https://imgur.com/TdlrvU9

Bibliografie/Resurse

https://www.arduino.cc/reference/en/libraries/servo/ https://ocw.cs.pub.ro/courses/pm/lab/lab3-2023

Export to PDF

Link Tinkercad + Code

https://www.tinkercad.com/things/dVtbHZf8xYY

#include <Servo.h>

Pin configuration const int temperaturePin = A0; Analog input pin for temperature sensor const int servoPin = 12; *Digital pin for servo control* Temperature threshold const float temperatureThreshold = 32.1; *Celsius* Servo parameters const int servoRotationAngle = 90; *Angle to rotate the servo const int servoInitialAngle = 0;* Initial angle of the servo const unsigned long servoDelay = 3000; *Delay in milliseconds for servo to stay at the rotated position const unsigned long servoRestDelay = 10000;* Delay in milliseconds for servo to rest at position 0

Servo servo; volatile bool isServoRotated = false; volatile bool isResting = false;

void setup() {

```
pinMode(temperaturePin, INPUT);
servo.attach(servoPin);
Serial.begin(9600);
```

```
// Set up Timer2
cli();
TCCR2A = 0;                             // Clear control register A
TCCR2B = 0;                             // Clear control register B
TCNT2 = 0;                              // Set initial value of the timer
counter
OCR2A = 156;                            // Set compare match value (1 Hz)
TCCR2A |= (1 << WGM21);                 // Turn on CTC mode
TCCR2B |= (1 << CS22) | (1 << CS21) | (1 << CS20);   // Set prescaler to
1024
TIMSK2 |= (1 << OCIE2A);                 // Enable timer compare interrupt
sei();
```

}

float readTemperature() {

```
int rawValue = analogRead(temperaturePin);
// Convert the raw value to temperature in Celsius
float temperature = ((float)rawValue / 1023.0) * 5.0;  // Convert the analog
value to voltage
temperature = (temperature - 0.5) * 100.0 / 2.95;              // Convert
voltage to Celsius
return temperature;
```

}

ISR(TIMER2_COMPA_vect) {

```
static unsigned long previousTime = 0;
unsigned long currentTime = millis();
unsigned long elapsedTime = currentTime - previousTime;
```

```
// Read temperature
float temperature = readTemperature();
Serial.println(temperature);
// Check if temperature is above threshold and servo is not currently
rotating or resting
if (temperature > temperatureThreshold && !isServoRotated && !isResting) {
  servo.write(servoRotationAngle);  // Rotate the servo
  isServoRotated = true;
  previousTime = currentTime;
}

// Check if the servo has reached the delay duration
if (isServoRotated && elapsedTime >= servoDelay) {
  servo.write(servoInitialAngle);    // Move the servo back to position 0
```

```
  isServoRotated = false;
  isResting = true;
  previousTime = currentTime;
}

// Check if the resting duration has passed
if (isResting && elapsedTime >= servoRestDelay) {
  isResting = false;
  previousTime = currentTime;
}
```

}