Документация

Борил Диянов Игнатов – 0MI0600124

2-ри курс – Софтуерно инженерство

СЪДЪРЖАНИЕ

ЗАДАНИЕ

Целта на проекта е да се създаде интелигентен домашен барман. Барманът робот ще позволи по-лесен достъп до набор от напитки, дали те ще са домашно направени или в истински барове предлагани. Този робот барман ще улесни достъпа на много потребители до любимата им напитка и предлага функционалност за самообслужване. Също така роботът позволява да се миксират ограничен брой напитки, като се прави проверка дали чашата ще прелее използвайки датчик за натоварване със сензори.

Роботът работи, като чрез перисталтична помпа изпомпва съдържанието на напитката от бутилката в чашата. За потребителски интерфейс се използва 16x2 LCD дисплей, който дава достъп до комуникация и улеснени на потребителя при работа с робота. За определение на размера на чашата, в която ще се налива, се използва датчик за натоварване. Приема се, че има два вида стандартизирани размери за чаши. С датчика ще се използва 24-bit ADC, който използва HX711 чип. Причини за това решение са, че се използва в много проекти с Ардуино и има съществуващи библиотеки относно този чип. Напрежението, което се захранва микроконтролера, идва от обикновено 5-волтово зарядно за мобилно устройство, като е свързано за USB порта на Ардуино. Мощността за самият робот включва превключващ регулатор за ефективно преобразуване на електрическата енергия, което позволява захранване с импулсен режим (SMPS).

БЮДЖЕТ

Електрически компоненти

- 1x Arduino UNO REV3 микроконтролер
- 1x Delta Electronics 12V 100W електрозахранващо устройство
- 2x Sanyo Denki 24V 0,48Nm стъпков мотор
- 2x Mikroe A4988 35V 2A моторен драйвер
- 4x CHERRY MX black бутони
- 1x Seeed Studio 16x2 LCD екран
- 1x JOY-IT 1kg HX711 датчик
- 1x Antec AT-12/SC 120mm охлаждащ вентилатор
- 1x C&K ключ
- SANMOTION кабели
- 1x Qualtek USB кабел

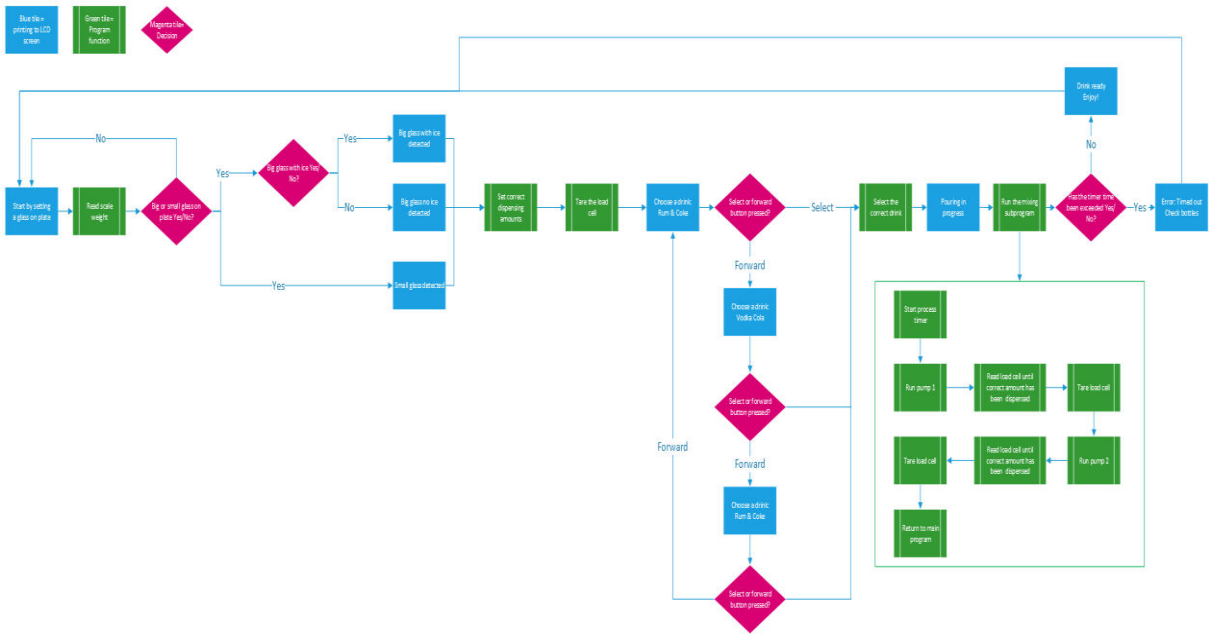
Механични компоненти

- Перисталтична помпа
- Лагери
- Силиконови тръби
- Различни хардуерни компоненти като болтове, гайки, пружини и други.

Среди за разработка

- Arduino Integrated Development Environment (IDE) v1
- Arduino IDE 1 on Windows

БЛОК-СХЕМА



Блок-схемата описва логиката зад машината, като последователно демонстрира всичките възможни операции, които машината е способна да изпълни. Машината започва работа веднага щом датчика засече присъствието на стъклената чаша върху плота на машината, след което започва да изчислява размерите на самата чаша, за да прецени какво количество от напитката да изсипе в нея. Една чаша е „голяма“, ако съдържа лед в себе си. Това означава, че машината трябва да изчисли какво количество течност да изсипе през диспенсъра. Софтуерът на барманът настройва необходимото количество според събраните данни и премахвайки тежестта на датчика за натоварване за точност на изчисленията. Предстои да се избере напитката от тези, които ще се покажата на LCD-екрана, с натискането на бутона “Forward”, за да се появи следващата напитка или бутона “Select”, за да се избере текущата напитка и да се изсипе в чашата. След изпълнението на тези функции идва ред на програмата да започне миксирането на напитката, като започва да отчита времето за напълване на чашата с избраната от потребителя напитка. Това се изпълнява заедно със сензорите на датчика, които изчисляват обема на чашата, за да измерят обема на чашата, за да не прелее. Активира се втората помпа, която изсипва втората напитка към чашата, за да се смесят,

където отново се проверява дали чашата ще прелее, като се използват отново сензорите на датчика за натоварване, който се намира на плота, върху който стои самата чаша за наливане. Ако времето се просрочи, т.е. отнема много време да се „напълни“ чашата, това означава, че ще трябва да се проверят ботилките с напитките, които се изсипват от машината в чашата, дали не са празни. Това означава, че ще трябва да се рестартира процесът отначало. В противен случай, на екранът ще се изпише, че напитката е готова и може да се консумира.

АЛГОРИТЪМ

Библиотеката `Wire.h` съдържа огромна част от функциите, които ще се използват за разработката на кода.

Библиотеката на *Seeed Studio* `rgb_lcd.h` съдържа функции, които помагат на функционирането на LCD дисплея да бъде по-удобно за потребителя. Основно тази библиотека придава чар на дисплея с многото функции, които съдържа.

Библиотеката `HX711.h` придава интерфейс/свързване с *Avia Semiconductor HX711 24-Bit Analog-to-Digital Converter (ADC)*.

```
#include <Wire.h>

#include "rgb_lcd.h"

#include "HX711.h"
```

Използваме следните глобални променливи, за да улеснят писането на кода, като предотвратяват повтарянето им в дефинициите на функциите, в които те ще се използват.

```
HX711 loadcell;

rgb_lcd lcd;

int n = 0;

volatile byte state = HIGH;

int run_pump = 0;
```

Глобални константи, които обособяват номера на пина на микроконтролера, но могат да се изберат други пинове на тяхно място. Това може да се изпълни ръчно чрез промяната на самият код, а не повреме на изпълнение на програмата.

```
const int LOADCELL_DOUT_PIN = 7;
```

```
const int LOADCELL_SCK_PIN = 8;
const int dirPin = 10;
const int stepPin = 11;
```

Настройва се датчиковия разделител, като се калиброва на удобно тегло под подразбиране. Задаваме количеството

```
long LOADCELL_DIVIDER = 1680;
float amount = 0;
int isBig;
float reading;
int enable_pin = 5;
```

Функцията setup() се изпълнява еднократно в началото и задава режима на пиновете и инициира серийна комуникация.

```
void setup() {
```

Настройват се пиновете на микроконтролера, както и обекта `Serial` позволява комуникация между микроконтролера и други устройства като компютъра. Настройва се режим за работа на съответните пинове.

```
  Serial.begin(9600);
  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
  pinMode(13, OUTPUT);
  pinMode(stepPin, OUTPUT);
  pinMode(dirPin, OUTPUT);
  pinMode(enable_pin, OUTPUT);
```

За да избегнем проблеми със засичането на времето ние използваме следните две функции за съответните пинове, по който начин постигаме автоматност на работата на микроконтролера. Изпълнението на всяка една ISR функция става последователно, като не може да се изпълняват две едновременно.

```
  attachInterrupt(digitalPinToInterrupt(2), change, LOW);
```

```
  attachInterrupt(digitalPinToInterrupt(3), choose, LOW);
```

Инициализира се LCD дисплея, като параметрите на функцията `begin(cols, rows)` посочват бройката колони и редове, които дисплеят има, след което функцията `clear()` изчиства екрана, което позволява да може да започне работа.

```
  lcd.begin(16, 2);
```

```
  lcd.clear();
```

Настройваме посоката на въртене на моторите по часовниковата стрелка или обратно на часовниковата стрелка, след което функнцията `scale_setup()` започва да отмерва размерите на чашата.

```
  digitalWrite(dirPin, LOW);
```

```
  scale_setup();
```

```
}
```

Функцията loop() се изпълнява непрекъснато – до спирането на захранването. В нея се извършват всички промени по състоянието на робота.

```
void loop() {
```

```
  float value = loadcell.get_value() /
LOADCELL_DIVIDER;
  Serial.println(value);
  if (value > 222 && value < 232){
    isBig = 1;
    run_lcd();
  }
  else if (value > 258 && value < 268){
    isBig = 0;
    run_lcd();
  }
  else {
    run_pump = 0;
    lcd.clear();
    lcd.print("Place the glass ");
    lcd.setCursor(0,1);
    lcd.print("on the platform");
  }
  delay(1000);
}
```

Използваме функция, която да активизира екрана и да може да се използва от потребителя

```
void run_lcd(){
  if (run_pump == 0) {
```

Стартира екрана, за да може потребителят да избере коя напитка да се налее в съда. Ако няма посочен номер на напитката от потребителя, екранът се изчиства и известява потребителя да си избере напитка, която барманът да му сипе.

```
    if (n == 0) {
      lcd.clear();
      lcd.print("Press 'select' ");
      lcd.setCursor(0,1);
      lcd.print("to start the machine!");
    }
```

Ако има вече посочен номер от потребителя, то системата ще посочи номерът на напитката, която ще бъде сипана в чашата и ще направи забавяне от 500ms, за да бъде избегнато блещукането на монитора повреме на сипване.

```
    else {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Number of drink: ");
      lcd.print(n);
      delay(500);
    }
  }
```

Когато помпата бъде подготвена за работа, на екрана ще пише, че може да започне процес на изсипване на течността в самата чаша, като се извика

функцията `pump_liquid(amount)`, където `amount` е необходимото количество, което трябва да бъде достигнато, за да бъде чашата пълна.

```
  else {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Is prepared!");
    lcd.print(n);
    pump_liquid(amount);
    delay(5000);
  }
}
```

Декларираме следната функция на адрес 0, която ще рестартира микроконтролера, когато бъде извикана `resetFunc()`.

```
void(*resetFunc) (void) = 0;
```

Използваме функцията

```
void change() {
  static unsigned long last_interrupt_time = 0;
  unsigned long interrupt_time = millis();
```

Ако настъпят някакви прекъсвания при повреме на работа на робота, които да надвишат 200ms, ще продължи процесът, защото това е просто bounce.

```
  if (interrupt_time - last_interrupt_time > 100) {
    n++;
    if (n > 4) {
```

```
      n=1;
    }
    last_interrupt_time = interrupt_time;
  }
}
```

Създаваме помощна функция, която позволява цикъла да достъпи функцията за помпане на напитката и избира колко количество от напитката да се изсипе в чашата.

```
void choose() {
    run_pump = 1;
    if (isBig == 1) {
      amount = 460;
    }
    else if (isBig == 0) {
      amount = 470;
    }
}
```

Наливането на алкохолни напитки се осъществява, като посочим определено количество, като когато то бъде достигнато, наливането приключва.

```
void pump_liquid(float amount) {
  Serial.print(amount);
  int accelerate = 2000;
```

За да ускорим процеса на работа на помпата ние създаваме цикъл, в който се декрементира времето за ускорение и така можем да избегнем потенциално преливане. С други думи, този цикъл осигурява внимателно наливане на напитката в чашата.

```
  for (int i = 0; i<= 350; i++){
    digitalWrite(enable_pin, HIGH);
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(accelerate);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(accelerate);
    accelerate = accelerate - 4;
  }
  reading = loadcell.get_value() / LOADCELL_DIVIDER;
  Serial.print(reading);
  int k = 0;
  while (reading < amount && reading > 1){
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(600);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(600);

    if (k > 100){
      reading = loadcell.get_value() /
LOADCELL_DIVIDER;
      k = 0;
    }
```

```
    k++;
  }
  digitalWrite(enable_pin, LOW);
  terminateProcess();
}
```

Когато напитката е готова, LCD екранът ще започне да блещука с честота от 2Hz, като изписва съобщение, че е време потребителят да отмести чашата от поставката с датчика. Щом бъде отместена чашата, променливата `reading` ще приеме стойност 0 както променливата `run_pump`, след което ще се рестартира системата с функцията `resetFunc()`.

```
void terminateProcess() {
  while (reading > 1) {
  lcd.clear();
  lcd.print("The drink is ready!");
  lcd.setCursor(0,1);
  lcd.print("You can take your drink.");
  reading = loadcell.get_value() / LOADCELL_DIVIDER;
  delay(500);
  }
  run_pump = 0;
  resetFunc();
}
```

Тази функция служи да се използва в началото, за да

```
void scale_setup() {
```

Започва да измерва теглото на чашата, като

```
 loadcell.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
```

Поставяме степента на калибровка за преуразмеряването на обекта във всяко едно измерение с тази степен, след което поставяме да е от степен 0.

```
  // loadcell.set_scale(LOADCELL_DIVIDER);
  loadcell.tare();
}
```

ЗАКЛЮЧЕНИЕ

Барманът робот изпълнява много елементарни функционалности и по този начин успява да създава коктейли от различни напитки, които ръчно се поставят във вътрешността на машината и се изсипват директно в чаша с установени размерности, които определят какво количество от напитката да бъде изсипено в нея, използвайки основно стъпкови мотори и перисталтични помпи директно в чашата и датчик за натоварване, който измерва тежестта на чашата.

Обаче има няколко подобрения, които биха усъвършенствали функционирането на машината и удобството на потребителя с нея:

- Стъпковите мотори създават много шум при работа, което би могло да се избегне с microstepping, т.е. да се настроят да не правят цяла стъпка при всяко едно движение.
- Текущата версия на бармана не съдържа втора перисталтична помпа, което означава, че потребителят трябва да е напълнил чашата си предварително с напитката, която да смеси.
- Предният панел трудно можеше да се постави върху шасито.
- Може да се замени платформата на чашата с решетка, на която да седи повреме на наливането на напитката в нея, за да се избегне потенциално преливане. Това обаче не би изглеждало козметически добре.

Цялостно, проектът изпълнява това, което се изисква основно от бармана, а то е да налива напитки интелигентно във всякакви чаши без да преливат, дори те да са вече пълни догоре.