universidade de aveiro

The Useless Box

Módulo: Microcontroladores e interação com sensores e atuadores

Realizado por:

Gonçalo Silva – 103244

Cláudia Silva –105076

Diana Garrido – 103784

Francisco Cardita –96640

Pedro Afonso – 104206

André Oliveira – 10285

Índice

Em que consiste a Useless Box

O nosso projeto baseia-se numa "brincadeira" muito comum, chamada de Useless Box. Na sua base, consiste num interruptor no topo de uma caixa e um braço que sai da caixa para desligar esse interruptor, quando este for ativado a caixa apresenta vários estados de espírito e níveis de agressividade.

Materiais que usamos

No projeto, usámos os seguintes materiais:

- 1 Buzzer
- 1 Servo para ativar o braço
- 1 Placa de desenvolvimento
- 1 interruptor "Single Pole Double Throw"
- 1 Placa branca
- 1 Display LCD I2C
- 1 Caixa de madeira

Comportamento da caixa

Primeira implementação, sem LCD

A caixa apresenta vários níveis de estado de espírito. Cada nível corresponde a uma ativação do interruptor e sempre que este é ativado, incrementa a agressividade até chegar ao nível máximo que é 10.

A cada nível estão associadas ações diferentes, em especial o barulho gerado pela caixa, vai aumentando consoante o nível até chegar a 7 e depois no 9.

No nível 8, a caixa começa a ficar zangada, então faz um barulho diferente, e quando esta desliga o interruptor fica algum tempo parada e a segurar no mesmo, para mostrar que não quer a nossa interação.

O nível 10 é semelhante ao nível 8, no entanto, neste o som produzido é diferente. Na sua essência, o som aumenta até um valor e depois diminui, parecido a uma sirene. A partir deste ponto, a caixa fica sempre no mesmo estado.

Implementação final, com LCD

A versão com display LCD funciona da mesma maneira, o único acrescento é a inclusão do LCD, a sua configuração, bem como interação.

Quando a caixa ainda não interagimos com a caixa, esta mostra no LCD que está a dormir com “ZzzZzzZZ”.

Em semelhança aos estados anteriores, no nível 8, a caixa começa a ficar extremamente zangada, na primeira linha do LCD mostra “PARA!” e na segunda “Deixa-me!”. O nível máximo, 10, é acompanhado da mensagem “Já chega, agora estou zangado!” na primeira linha e na segunda “Agora vais levar!”.

Nos restantes níveis, na primeira linha do LCD fica o texto “Estás-me a chatear” e na segunda, proporcionalmente ao nível de agressividade, o caracter “!”, por exemplo, no nível 3 de agressividade, a segunda linha do LCD mostra “!!!”.

Configuração e desenvolvimento

Configuração do Arduíno IDE STM32 F411RE

Vista que a placa STM32 F411RE consegue funcionar com código em Arduíno, optámos por seguir nesse rumo, devido à extensa documentação e adoção da programação em Arduíno. Para isso foi preciso primeiro configurar o IDE para comunicar corretamente com a placa.

Primeiro, dentro do Arduíno, é preciso fazer o download do pacote de placas STM32. Por isso, acedemos ao menu “Ferramentas” -> “Placas” -> “Gestor de placas” e procuramos pelo pacote “STM32 MCU based boards” e fazemos download.

Após este primeiro passo, uma grande gama de configurações para diversas placas STM32 ficará disponível na IDE, então agora é preciso selecionar as opções corretas relativamente à nossa placa.

No nosso caso, estamos a utilizar uma STM32 F411RE, então em “Ferramentas” -> “Placas” selecionamos “Nucleo-64”, em “Board part number” -> “Nucleo F411RE” e para “Upload method” a opção “SWD”.

Agora para que seja possível utilizar as funções “Serial” do Arduíno, ainda em “Ferramentas” -> “USB support” a opção “Nenhum”, de forma a ser possível fazer debug, ou seja, imprimir mensagens com informação da placa no nosso ecrã.

Finalmente, para que a IDE programe a placa, precisamos de fazer download e instalação do STM32CubeProgrammer.

Desenvolvimento do código

```
#include <Servo.h> // Include the Servo library 4
#include <Wire.h> // Librare used to locate the display
#include <LiquidCrystal_I2C.h>  // Used to interact with the display

const int servoPin = D8;
const int buzzerPin = D2; // Declare the buzzer
const int buzzerAuxPin = D4;
const int switchPin = D7;
int angryLevel = 0;
int lastValue = LOW;


Servo Servo1; // Create a servo object
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  Servo1.attach(servoPin);  // Bind pin to Servo
  pinMode(buzzerPin, OUTPUT); // Set buzzer as an output
  pinMode(buzzerAuxPin, OUTPUT);
  pinMode(switchPin, INPUT);  // Configure Pin to activate
  Serial.begin(9600);
  lastValue = -1; // Value to be used on the switch

  lcd.begin();  // initialize the LCD

}

void loop() {
  int sValue = digitalRead(switchPin);  // Read switch value
  Serial.print("Switch: ");
  Serial.print(sValue);
  Serial.print("; LastValue: ");
  Serial.println(lastValue);
  showScreen();
  if(sValue == HIGH && lastValue == 0){ // only operate if switch value was changed
    increaseAnger();
    closeButton();
    buzz();
  }else{
    hide();
  }

  if(lastValue != sValue) // find if switch was altered
    lastValue = sValue;
}


/* Manage Anger */
void increaseAnger(){
  if(angryLevel < 10)
    angryLevel++;
}
```

```
/* Update info on the screen */
void showScreen(){
  lcd.backlight();
  static int efect = 0; // static variable to produce effects

  if(angryLevel == 0){
     if(efect) // different lines to produce a sleeping result
      lcd.print("ZzzZZzZ!");
     else
      lcd.print("zZzzZZzz!");

     efect = !efect;
  }else if(angryLevel >=1 && angryLevel <= 7 || angryLevel == 9){
     lcd.clear();
     lcd.print("Estás-me a chatear!");
     lcd.setCursor(0, 1); // change lines
     int i;
     for(i=1;i<=angryLevel;i++)
      lcd.print("!");
  }
  else if(angryLevel == 8){
    lcd.clear();
    lcd.print("PARA!");
    lcd.setCursor(0, 1);
    lcd.print("Deixa-me!");
  }
  else if(angryLevel == 10){
    lcd.clear();
    lcd.print("Já chega, estou zangado!");
    lcd.setCursor(0, 1);
    lcd.print("Agora vais levar!");
  }
}


/* Turn off the switch */
void closeButton(){
  Servo1.write(0);
}

/* Hide the arm */
void hide(){
  Servo1.write(180);
}
```

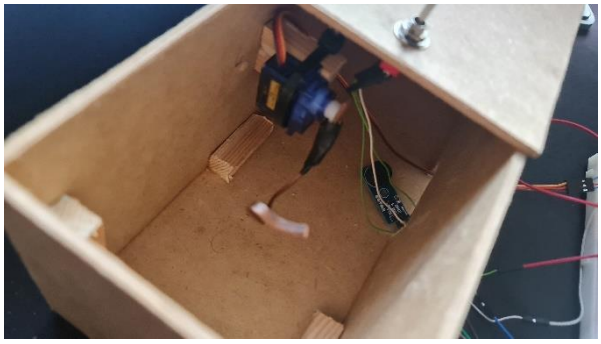
```
/* Buzz or Alert anger */
void buzz(){
  if(angryLevel <= 0) return; // pre-condition
  Serial.println(angryLevel);

  if(angryLevel < 8 || angryLevel == 9){
    tone(buzzerPin, (400*angryLevel));
    digitalWrite(buzzerAuxPin, HIGH);
    delay(500);
  }else if(angryLevel == 8){
    for(int i=1000 ; i<(2000 + angryLevel * 100) ; i+=angryLevel){ // loop for to increase the sound frequency
         tone(buzzerPin, i);
         digitalWrite(buzzerAuxPin, HIGH);
         delay(10); //You can change the delay if you want to make it longer or shorter
    }
  }else{ // angryLevel 10
    int i;
    for(i=1000 ; i<(3000 + angryLevel * 100) ; i+=angryLevel){ // loop for to increase the sound frequency
         tone(buzzerPin, i);
         digitalWrite(buzzerAuxPin, HIGH);
         delay(10); //You can change the delay if you want to make it longer or shorter
    }
    for(; i>1000 ; i-=angryLevel){ // loop for to increase the sound frequency
         tone(buzzerPin, i);
         digitalWrite(buzzerAuxPin, HIGH);
         delay(10); //You can change the delay if you want to make it longer or shorter
    }
  }
  noTone(buzzerPin);     // Stop sound...
  digitalWrite(buzzerAuxPin, LOW);

}
```

Construção da caixa

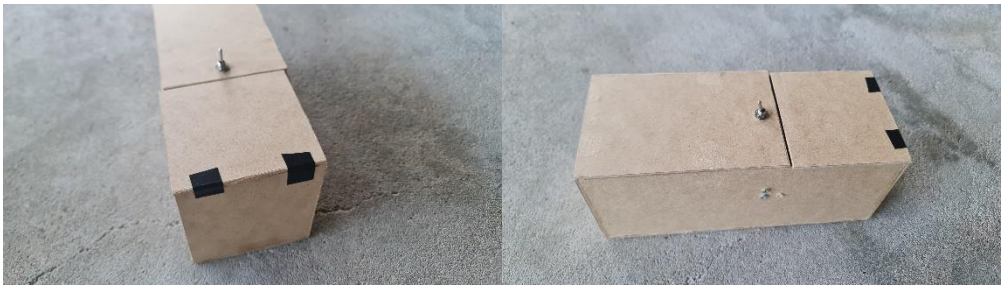
Para construção da caixa foi usada madeira MDF. Após a caixa construída, passámos para o mecanismo de desligar o botão. Foram experimentadas várias opções, mas a que acabámos por usar, foi uma simples metade de anilha de cobre que moldámos para que a mesma pudesse ter a amplitude e formato suficientes para desligar o botão.

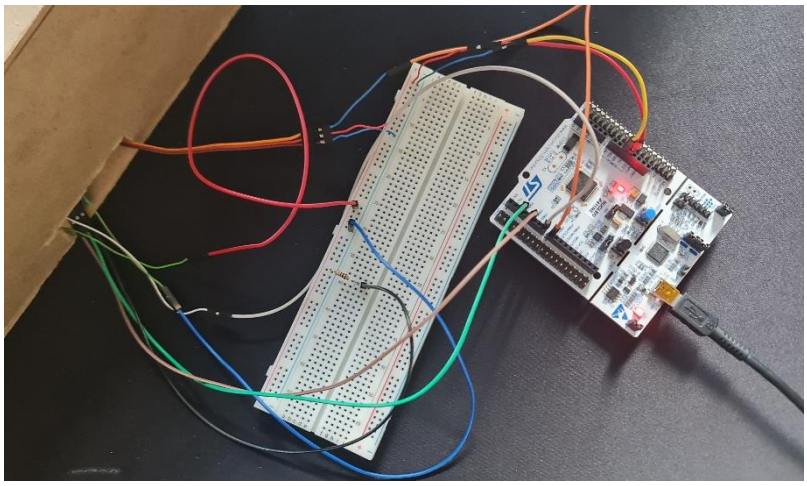
Agora faltava arranjar uma boa maneira de juntar o servo e a anilha moldada. Após algumas experimentações, optámos por um simples arame de porta-chaves e dois furos na anilha, para que estes se interligassem com os do servo e com um pouco de fita-cola isoladora para que estes arames não saíssem fora do sítio.

Estando o mecanismo feito, passámos a descobrir o melhor lugar para fixar o mesmo e mais uma vez, após tentativa e erro, acabámos por encontrar a posição certa para o fixar, de forma que este consiga operar sobre o botão.



Como o objetivo seria que o mecanismo abrisse uma tampa e desligasse o botão, colocámos a tampa. Mas visto a grossura da madeira não ser a ideal para colocar dobradiças, acabámos por colocar apenas dois pedaços de fita-cola que simulavam as dobradiças.





Aspetos finais e observações

Usar os três componentes ao mesmo tempo (servo, buzzer, LCD e switch), ligados a só um pino de alimentação de 5v, não é aconselhável, visto que a energia necessária para operar o servo, de forma a este ter potência suficiente para desligar o interruptor não é fornecida. O ideal seria usar outra placa e ligar o pin de 5v dessa placa ao servo, ou então, qualquer outro método de alimentação de energia, como uma pilha de 5v para alimentar o servo.

É da nossa opinião que os objetivos do projeto foram concluídos e conseguimos criar uma ideia bastante interessante. A demonstração do projeto (sem o LCD, por questões de energia), pode ser observada através do link.