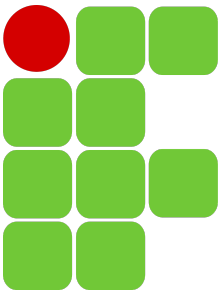


INSTITUTO FEDERAL DE EDUCAÇÃO CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO NORTE
CAMPUS - CANGUARETAMA

Projeto final Sistemas Embarcados - Sistema de Irrigação

Adriana Euflazino da Silva
Edson Domingos da Silva
Jobson Frank Floriano
Laura Kauanne da Silva
Nayonara Galvão da Silva
Oziele Rodrigues de Oliveira

Professor: Bruno Augusto Vitorino

1. Objetivo

O objetivo principal do nosso projeto foi baseado em montar um sistema para ligar e desligar uma torneira sem que haja a presença do contato humano, utilizando o arduino uno, um RTC (Relógio de Tempo Real), um servo motor, uma torneira e algumas conexões, como joelho e pedaços de cano.

2. Desenvolvimento

2.1. Material

Abaixo está os materiais que utilizamos e suas respectivas funções:

- Arduino - Controlar a rotação do servo motor, o que permite controlar a abertura da torneira;
- O Servo Motor - Fechar e abrir a torneira, conforme o tempo que seja determinado pelo relógio em tempo real;
- O RTC - Armazenar o tempo para ligar e desligar a torneira;
- E a torneira - será o que vai ser controlado;
- Conexões de canos - Ajudar a segurar o motor.

2.2. Código

Abaixo está o código utilizado para o funcionamento do projeto por meio da *IDE* do arduino:

```
#include <ThreeWire.h> //INCLUSÃO DA BIBLIOTECA
#include <RtcDS1302.h> //INCLUSÃO DA BIBLIOTECA
#include <Wire.h>
#include <Servo.h>

Servo myservo;
int pos = 0;

ThreeWire myWire(3, 4, 2); //OBJETO DO TIPO ThreeWire
RtcDS1302<ThreeWire> Rtc(myWire); //OBJETO DO TIPO RtcDS1302

void setup () {
```

```
    Serial.begin(9600); //INICIALIZA A SERIAL
    myservo.attach(9);
    Rtc.Begin(); //INICIALIZA O RTC
      Serial.print("Compilado em: "); //IMPRIME O TEXTO NO
MONITOR SERIAL
     RtcDateTime compiled = RtcDateTime(__DATE__, __TIME__);
//VARIÁVEL RECEBE DATA E HORA DE COMPILAÇÃO
      printDateTime(compiled); //PASSA OS PARÂMETROS PARA A
FUNÇÃO printDateTime
    Serial.println(); //QUEBRA DE LINHA NA SERIAL
    Serial.println(); //QUEBRA DE LINHA NA SERIAL

        if(Rtc.GetIsWriteProtected()){  //SE O RTC ESTIVER
PROTEGIDO CONTRA GRAVAÇÃO, FAZ
         Serial.println("RTC está protegido contra gravação.
Habilitando a gravação agora..."); //IMPRIME O TEXTO NO
MONITOR SERIAL
         Rtc.SetIsWriteProtected(false); //HABILITA GRAVAÇÃO
NO RTC
        Serial.println(); //QUEBRA DE LINHA NA SERIAL
    }

       if(!Rtc.GetIsRunning()){  //SE RTC NÃO ESTIVER SENDO
EXECUTADO, FAZ
          Serial.println("RTC não está funcionando de forma
contínua. Iniciando agora..."); //IMPRIME O TEXTO NO MONITOR
SERIAL
        Rtc.SetIsRunning(true); //INICIALIZA O RTC
        Serial.println(); //QUEBRA DE LINHA NA SERIAL
    }

     RtcDateTime now = Rtc.GetDateTime(); //VARIÁVEL RECEBE
INFORMAÇÕES
```

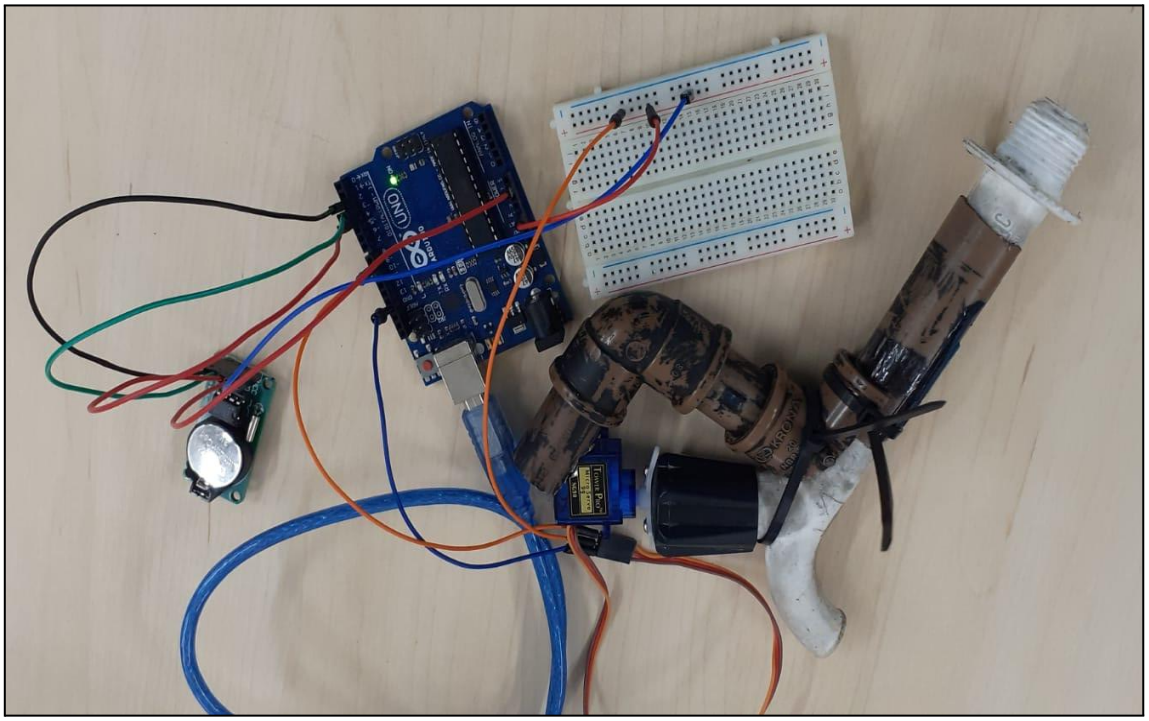
```
      if (now < compiled) { //SE A INFORMAÇÃO REGISTRADA FOR MENOR QUE A INFORMAÇÃO COMPILADA, FAZ
          Serial.println("As informações atuais do RTC estão desatualizadas. Atualizando informações..."); //IMPRIME O TEXTO NO MONITOR SERIAL
          Rtc.SetDateTime(compiled); //INFORMAÇÕES COMPILADAS SUBSTITUEM AS INFORMAÇÕES ANTERIORES
        Serial.println(); //QUEBRA DE LINHA NA SERIAL
    }
      else if (now > compiled){ //SENÃO, SE A INFORMAÇÃO REGISTRADA FOR MAIOR QUE A INFORMAÇÃO COMPILADA, FAZ
        Serial.println("As informações atuais do RTC são mais recentes que as de compilação. Isso é o esperado."); //IMPRIME O TEXTO NO MONITOR SERIAL
        Serial.println(); //QUEBRA DE LINHA NA SERIAL
    }
      else if (now == compiled) { //SENÃO, SE A INFORMAÇÃO REGISTRADA FOR IGUAL A INFORMAÇÃO COMPILADA, FAZ
          Serial.println("As informações atuais do RTC são iguais as de compilação! Não é o esperado, mas está tudo OK."); //IMPRIME O TEXTO NO MONITOR SERIAL
        Serial.println(); //QUEBRA DE LINHA NA SERIAL
    }
}

void loop () {
     RtcDateTime now = Rtc.GetDateTime(); //VARIÁVEL RECEBE INFORMAÇÕES
     printDateTime(now); //PASSA OS PARÂMETROS PARA A FUNÇÃO printDateTime
    Serial.println();
    delay(1000); //INTERVALO DE 1 SEGUNDO
```

```
     if (now.Hour() == 19 & now.Minute() == 02 & now.Second()
== 00) {
    Serial.println("ABRIU");
     for (pos = 180; pos >= 0; pos -= 1) { // goes from 180
degrees to 0 degrees
      myservo.write(pos);              // tell servo to go to
position in variable 'pos'
      delay(5);
    }
  }
  if (now.Hour() == 19  & now.Minute() == 02  & now.Second()
== 10){
    Serial.println("FECHOU");
     for (pos = 0; pos <= 180; pos += 1) { // goes from 0
degrees to 180 degrees
      // in steps of 1 degree
      myservo.write(pos);              // tell servo to go to
position in variable 'pos'
      delay(5);                        // waits 15 ms for the
servo to reach the position
    }
  }
}

#define countof(a) (sizeof(a) / sizeof(a[0]))

void printDateTime(const RtcDateTime& dt){
     char datestring[20]; //VARIÁVEL ARMAZENA AS INFORMAÇÕES
DE DATA E HORA

    snprintf_P(datestring,
            countof(datestring),
```

```
            PSTR("%02u/%02u/%04u %02u:%02u:%02u"), //FORMATO
DE EXIBIÇÃO DAS INFORMAÇÕES
            dt.Day(), //DIA
            dt.Month(), //MÊS
            dt.Year(), //ANO
            dt.Hour(), //HORA
            dt.Minute(), //MINUTOS
            dt.Second() ); //SEGUNDOS
    Serial.print(datestring); //IMPRIME NO MONITOR SERIAL AS
INFORMAÇÕES
}
```

3. Resultado

Figura 1: Protótipo do projeto torneira automatizada.



Fonte: Do próprio autor (2023)

4. Conclusão

O resultado final foi o esperado, tendo em vista que está tudo funcionando de forma correta. O servo motor rotaciona 180 graus a torneira, que é aberta, quando a hora, minuto e segundo é chegado de forma exata. A hora é recebida por meio do RTC que é transmitida para o arduino uno e passada o comando rotação para o servo motor. O mesmo se dá ao rotacionar 180 graus de forma contrária.

Repositório do projeto no github:
https://github.com/Edsondomingos/onOff-controle-torneira-arduino