Disciplina Software Embarcado

Curso Mestrado em Redes de Computadores

Prof. Francisco Sant'Anna

Daniel Luzente de Lima

Disciplina Software Embarcado

Projeto Final

Sistema de Automatização para Ambiente de Datacenter

Disciplina Software Embarcado

Objetivo

Automatizar e otimizar controle de iluminação e de refrigeração em ambiente de Datacenter.

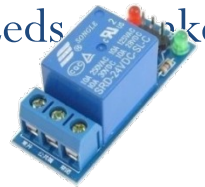
Disciplina Software Embarcado

Componentes do Sistema

Sensores – Botão, AM2302 (Temperatura e umidade), Flex Sensor (Flexão e Deflexão) e PIR Sensor (Movimento)



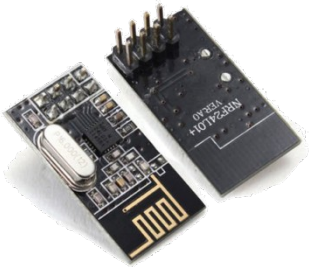

Atuadores – Leds, Buzzer e Módulo Relé



Disciplina Software Embarcado

Componentes do Sistema - Cont.

Comunicação RF - NRF24L01 transceiver module



Disciplina Software Embarcado

Componentes do Sistema - Cont.

Microcontroladores – 2 (dois) Arduinos Uno

RX ← TX

Disciplina Software Embarcado

Controles do sistema

Controle

Ativação da iluminação inicial do ambiente ao pressionar botão e abrir a porta.

Esquema lógico

Capturar pressionamento do botão abrir e o nível de flexão do sensor que estará acoplado à porta e enviar os dados ao micro controlador responsável pelos atuadores.

Disciplina Software Embarcado

Controles do sistema

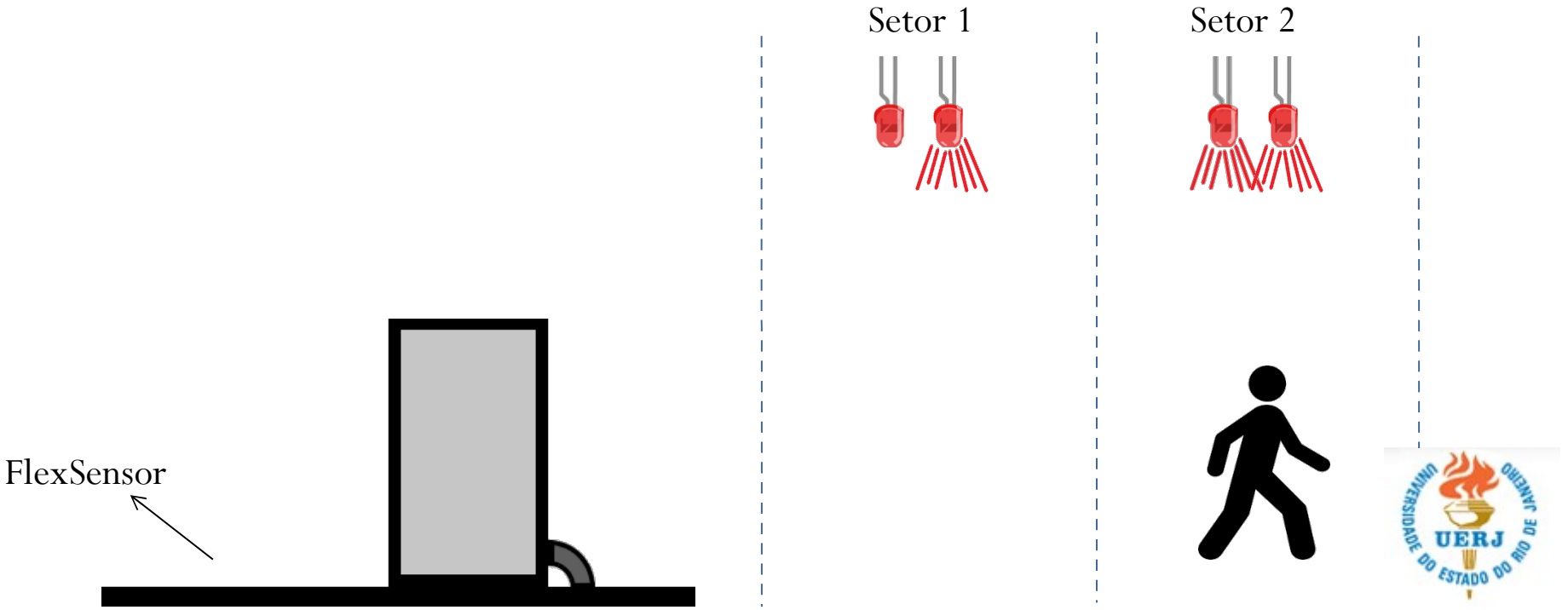
Controle

Ativação da iluminação setorial do ambiente baseado na detecção de localização e movimento do visitante.

Esquema lógico

Capturar dados de localização do visitante e enviar ao micro controlador para que se ative iluminação dupla no setor e iluminação única nos demais setores.

Disciplina Software Embarcado



Disciplina Software Embarcado

Controles do sistema

Controle

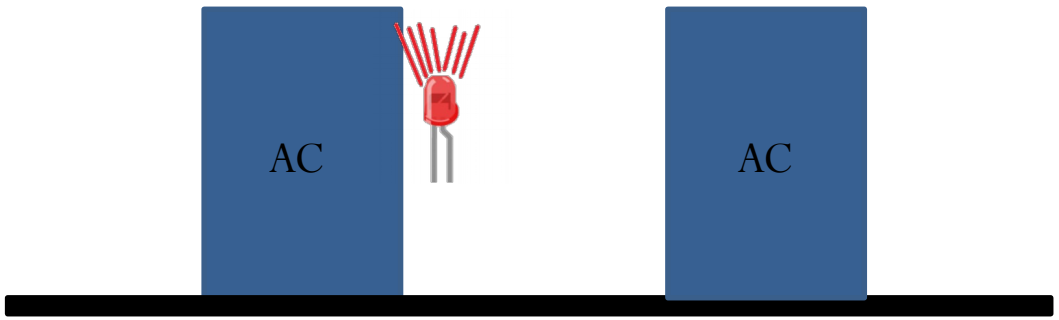
Ajustar potência da refrigeração do ambiente de acordo com a temperatura no momento e utilizar relés para acionamento.

Esquema lógico

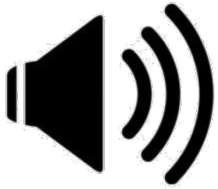
Capturar dados de temperatura e enviar ao micro controlador para que seja feita a regulação da potência da refrigeração.

Disciplina Software Embarcado

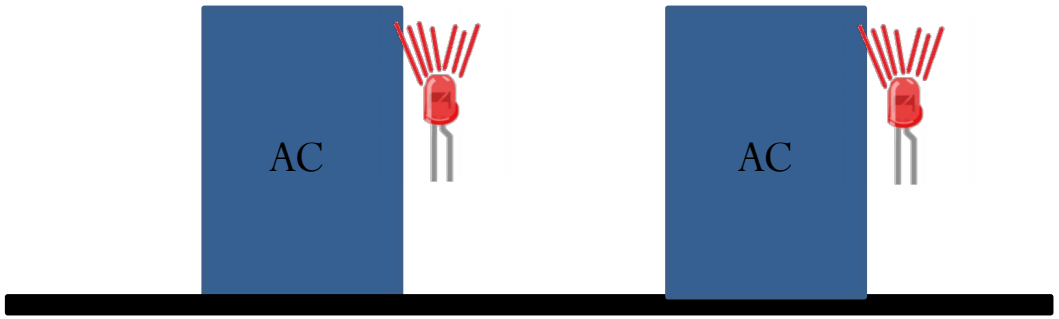
Temperatura: 20º



Disciplina Software Embarcado



Temperatura: 26º



Disciplina Software Embarcado

Controles do sistema

Controle

Desativação da iluminação do ambiente baseado na detecção de saída do visitante.

Esquema lógico

Capturar pressionamento do botão fechar, o nível de inclinação da porta e enviar os dados ao micro controlador responsável pelos atuadores.

Disciplina Software Embarcado

Status do Projeto

- Sensores de temperatura e de flexão configurados
- Transmissão RF funcionando
- Recebimento dos dados e atuação dos dispositivos funcionando
- Sensor de presença e iluminação setorial funcionando
- Comandos para ativação de refrigeração adicional funcionando
- Alarme de temperatura alta funcionando
- Ativação da refrigeração através de relés em fase de configuração
- Uso de interrupção em fase de configuração

Disciplina Software Embarcado

Código-Fonte

Disciplina Software Embarcado

```
#include <avr/interrupt.h>      //Biblioteca para interrupções
#include <SPI.h>                //Biblioteca para comunicação serial
#include <nRF24L01.h>           //Biblioteca para rádiofrequência
#include <RF24.h>               //Biblioteca para rádiofrequência
RF24 radio(7, 8);               // CE, CSN
#include <DHT.h>                //Biblioteca do sensor DHT
#define DHTPIN 4                //Sensor DHT no pino 2
#define DHTTYPE DHT22           // DHT 22  (AM2302)
DHT dht(DHTPIN, DHTTYPE);       // Initialize DHT sensor for normal 16mhz Arduino

const uint64_t pipe = 0xF0F0F0F0F0E1;  //Canal único para transmissão via rádi
```

Disciplina Software Embarcado

```
//Dataset para armazenar dados dos sensores que serão transmitidos via RF
typedef struct{
  float tempReading; //Stores temperature value
  int flexSensorReading;
  int havePeople;
  int haveNotPeople;
  int valorSensorPIR;
}
A_t;

A_t dataset;
```

Disciplina Software Embarcado

```
void setup()
{
  radio.begin();                        //Inicializa funções de rádio
  radio.setDataRate( RF24_250KBPS );    //Limita taxa de transferência em 250KBPS

  dht.begin();                          //Inicializa funções do sensor DHT

  pinMode(pinSensorPIR,INPUT);          //Sensor de presença
  pinMode(havePeopleBtn, INPUT);        //Botão de entrada com interrupção
  EICRA = (1 << ISC11);                 //INT1 modo RISING
  EIMSK = (1 << INT1);                  //Habilita external interrupt INT1
  pinMode(haveNotPeopleBtn, INPUT);     //Botão de saida sem interrupção

  radio.openWritingPipe(pipe);          //Abre canal de rádio para escrita
```

Disciplina Software Embarcado

```
//Interrupção do botão de entrada
ISR(INT1_vect)
{
  if ((millis() - lastDebounceTime) > debounceDelay) {
    readingBtn1 = 1;
  }
}
```

Disciplina Software Embarcado

```
if(dataset.tempReading >= 32.60 && dataset.tempReading < 32.70){
  digitalWrite(LED_AC_2, LOW);
  digitalWrite(LED_AC_1, HIGH);
  tone(speakerPin, 4978, 250);
  delay(1500);
  noTone(speakerPin);
} else if (dataset.tempReading >= 32.80){
  digitalWrite(LED_AC_1, HIGH);
  digitalWrite(LED_AC_2, HIGH);
  tone(speakerPin, 4978, 250);
  delay(500);
  noTone(speakerPin);
} else {
  digitalWrite(LED_AC_1, LOW);
  digitalWrite(LED_AC_2, LOW);
}
```

Disciplina Software Embarcado

```
if(dataset.flexSensorReading<=350 && doorOpen == 1 && lightFull == 0){
  lightFull = 1;                        //Seta variável para ele não entrar nesse If novamente
  digitalWrite(LED_SETOR1_1, HIGH);  //Liga iluminação total do ambiente
  digitalWrite(LED_SETOR1_2, HIGH);
  digitalWrite(LED_SETOR2_1, HIGH);
  digitalWrite(LED_SETOR2_2, HIGH);
  delay(5000);
  digitalWrite(LED_SETOR1_1, LOW);    //Desliga metade da iluminação do ambiente após 5 segundos
  digitalWrite(LED_SETOR2_2, LOW);
}
```

Disciplina Software Embarcado

```
if (dataset.valorSensorPIR == 1 && doorOpen == 1 && lightFull == 1){
  previousMillisPIR = millis();
  digitalWrite(LED_SETOR1_1, HIGH);
}

unsigned long currentMillisPIR = millis();
if (dataset.valorSensorPIR == 0 && doorOpen == 1 && lightFull == 1 && currentMillisPIR - previousMillisPIR >= interval2){
  digitalWrite(LED_SETOR1_1, LOW);
}
```

Disciplina Software Embarcado

```
    if(dataset.haveNotPeople == 1){
        doorOpen = 0;
        lightFull = 0;
        digitalWrite(LED_SETOR1_1, LOW);
        digitalWrite(LED_SETOR1_2, LOW);
        digitalWrite(LED_SETOR2_1, LOW);
        digitalWrite(LED_SETOR2_2, LOW);
      }
}
```

Disciplina Software Embarcado

FIM