PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE CIÊNCIAS EXATAS E DA COMPUTAÇÃO

VITOR DE ALMEIDA SILVA
Matricula: 2016.1.0033.0549-7
HIGOR ALVES FEREIRA
Matrícula: 2016.1.0033.0543-8

TRABALHO 2 N2: CONTROLE DE RECURSOS (IRRIGAÇÃO PARA HORTA)

TALLES MARCELO G DE A BARBOSA

GOIÂNIA,
2019

VITOR DE ALMEIDA SILVA
Matricula: 2016.1.0033.0549-7
HIGOR ALVES FERREIRA
Matrícula: 2016.1.0033.0543-8

TRABALHO 2 N2: CONTROLE DE RECURSOS (IRRIGAÇÃO PARA HORTA)

Trabalho apresentado como requisito parcial para obtenção de nota na disciplina Sistemas Embarcados no Curso de Engenharia da computação, na Pontifícia Universidade Católica de Goiás.
Talles Marcelo G de a Barbosa

GOIÂNIA,
2019

1 ENUNCIADO

Escolher algum protocólo de comunicação entre microcontroladores, por exemplo, Comunicação em anel ou Barramento e implementar algum caso de comunicação, por exemplo:

- **Eleição de lider;**
- **Exclusão mútua;**
- **Sincronização de tempo;**
- **Aloação de recursos;**
- **... entre outros**

Implementar utilizando no mínimo 3 arduinos, de modo a estabelecer comunicação entre eles.

2 DESENVOLVIMENTO

2.1 Teoria Alocação/Gerência de recursos

O tema definido foi **alocação/Gerência de recursos.** A principal característica de alocação de recursos é a de tratar do aumento da demanda de recursos distribuindo-os entre as tarefas concorrentes. Voltando para gerência de recursos, um sistema tal que gerência recursos precisa não só tê-los disponíveis (desempenho) como também conseguir disponibilizar estes recursos (escalonamento) (COULOURIS et al., 2013).

Em relação ao escalonamento dos recursos, uma estratégia simples é aplicar o escalonamento em ródízio (*round-robin*) nos fluxos. Para isso, dada uma taxa de quadros é possível calcular quando cada pacote deve ser enviado completamente. Desse modo existem técnicas de escalonamento clássicas, onde entre elas se encontram o EDF (*early deadline first*) e o RM (*Rate-Monotônic*) (COULOURIS et al., 2013).

O EDF exige que se tenha um escalonamento por mensagem, porém, seria mais eficiente basear o escalonamento nos elementos que existem por um tempo maior. O RM é uma técnica vinda para escalonar em tempo real de processos periódicos, que obtêm exatamente isso (COULOURIS et al., 2013). O RM foi escolhido para o problema em questão.

2.2 Contextualização

De forma simplificada, deseja-se montar um sistema de irrigação para uma plantação de verduras de dois tipos diferentes. O Tipo de verdura A precisa ser irrigada por um total de 200ms a cada 300ms, o tipo B precisa ser irrigada por um total de 100ms a cada 300ms de espera. as verduras são separadas em 2 canteiros diferentes, de tal forma que eles são bem distantes um do outro, porém, não o suficiente para atrapalhar a comunicação com fios de barramento.

Um agricultor (este é você) que tem como robe montar sistemas embarcados, tem em sua residência um total de 3 Arduinos sem uso algum. Ao ver isso, ele teve a ideia de utilizar todos os 3 Arduinos e montar um sistema de irrigação para suas verduras A e B. A ideia é montar o sistema de tal forma que 2 Arduinos fiquem responsável por um aspersor cada, e um terceiro Arduino

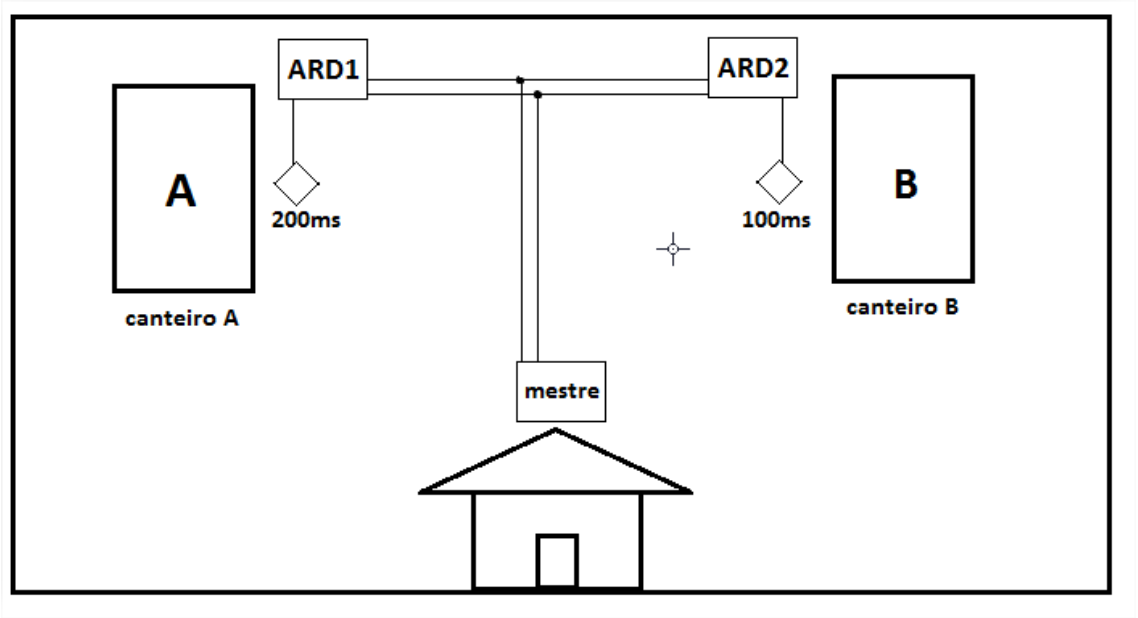
(presente na residência) dite aos outros 2 o tempo de irrigação de cada canteiro (aciona um de cada vez).

2.3 Solução

Para solucionar este problema, foi associado a cada tipo de verdura um Arduino. Como não existem restrições acerca do clima ou humidade, somente em relação ao tempo de uso do recurso (neste exercício o recurso é o aspersor), foi montado um sistema mestre/escravo com o mestre gerenciando a distribuição dos tempos dos recursos e os escravos utilizando esse tempo.

Para montar tal sistema, foi decido utilizar o protocolo I2C de forma a conectar todos os arduinos ao barramento. Cada Arduino terá um endereço, este será utilizado pelo mestre para mandar o tempo de uso do recurso. A Figura 1 mostra o esquema de ligação dos arduinos no ambiente definido.

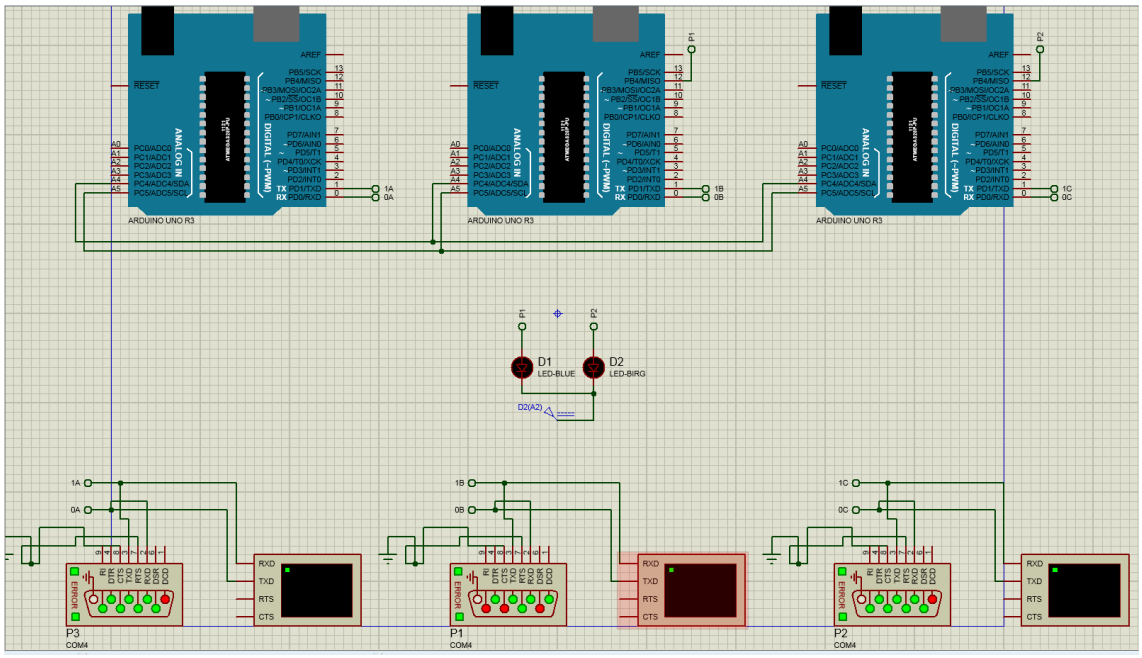
Figura 1: Ambiente



Fonte: Autoral

A Figura 1 é apenas um modelo ilustrativo. A simulação do Sistema foi feita no Proteus, no lugar dos aspersores foram utilizados leds, somente para simular um sinal de ativação (um modelo real iria precisar de equipamentos como relê, fontes externas e outros). Os losangos são os aspersores. Os terminais irão mostras as distribuições de recursos. A Figura 2 mostra a topologia montada no Proteus.

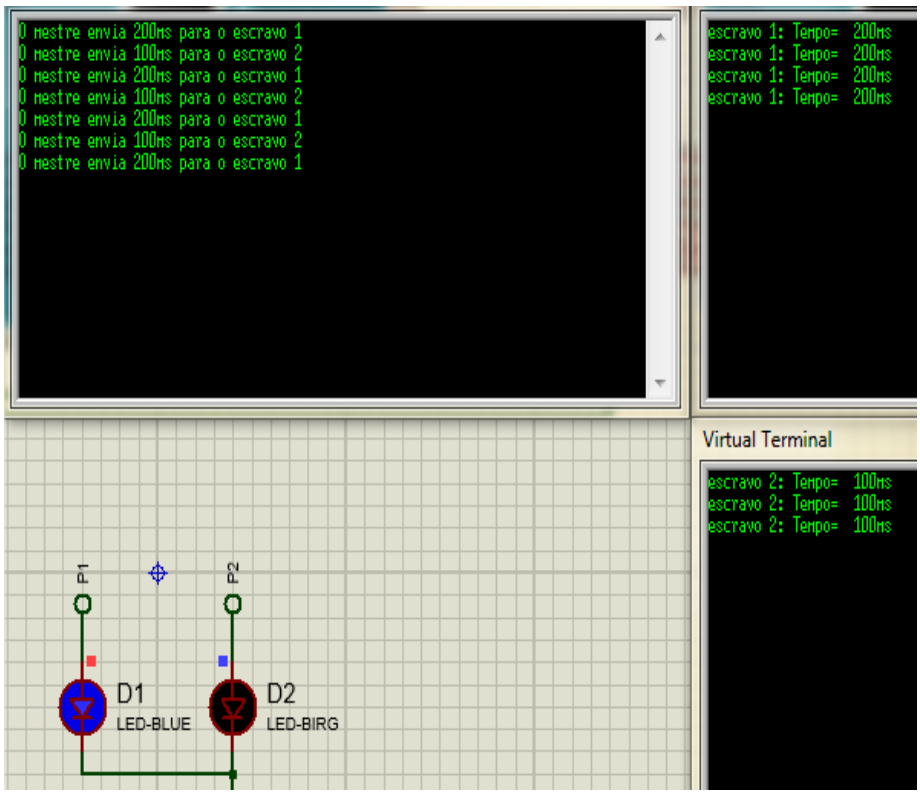
Figura 2: Topologia no Proteus



Fonte: Autoral

A Figura 3 mostra a distribuição dos recursos pelo Arduino mestre.

Figura 3: distribuição dos recursos



Fonte: Autoral

Dessa forma foi realizada a montagem do sistema requerido pelo exercício. “as verduras A e B agora poderão crescer bem saldáveis :D”.

Os códigos são mostrados nas Figuras 4 e 5.

Figura 4: Código do mestre

```
#include <Wire.h>
#define Slave1 200
#define Slave2 100
#define pin1 12


void setup()
{
  Wire.begin(); // join i2c bus (address optional for master)
  Serial.begin(9600);           // start serial for output
}

byte x = 0;


void loop()
{

  Serial.print("O mestre envia ");
  Serial.print(Slave1);
  Serial.println("ms para o escravo 1 ");
 // x++;
  Wire.beginTransmission(1);     // transmite para o 1
  Wire.write("Tempo=  ");        // envia 5 bytes
  Wire.write(Slave1);            // envia 1 byte
  Wire.endTransmission();        // encerra a transmissão

  delay(Slave1 + 100);           //delay para aguardar o uso do recurso pelo slave

  Serial.print("O mestre envia ");
  Serial.print(Slave2);
  Serial.println("ms para o escravo 2 ");

  Wire.beginTransmission(2);     //transmite para 2
  Wire.write("Tempo=  ");
  Wire.write(Slave2);
  Wire.endTransmission();

  delay(Slave2 + 100);

}
```

Fonte: Autoral

Figura 5: Código escravo 1

```
#include <Wire.h>
#define pin1 12


void setup()
{
  Wire.begin(1);                // join i2c bus with address #4
  Wire.onReceive(receiveEvent); // register event
  Serial.begin(9600);           // start serial for output
  pinMode(pin1 , OUTPUT);
}

//---
int s = 0; //variável que controla o momento de utilizar o recurso
int tempo = 0;  //tempo que será recebido do mestre para utilizar o recurso

void loop()
{
  //delay(100);
  // --- if de controle do recurso
  if (s == 1){
    usarLed();
    s = 0;
    tempo=0;
  }

}

// function that executes whenever data is received from master
// this function is registered as an event, see setup()


//void usarLed(int tempo)
void usarLed()
{
  digitalWrite(pin1 , HIGH);
  delay(tempo);
  digitalWrite(pin1 ,LOW);
  delay(tempo);
}


void receiveEvent(int howMany)
{
  //delay (100);

  Serial.print("escravo 1: ");
  while(1 < Wire.available()) // loop through all but the last
  {
    char c = Wire.read(); // receive byte as a character
    Serial.print(c);         // print the character
  }
  int x = Wire.read();    // receive byte as an integer
  //Serial.print("slave response: ");
  Serial.print(x);         // print the integer
  Serial.println("ms ");

  //usarLed(x);

  // ---
  s = 1;  //seta uso do recurso
  tempo = x;  //tempo de uso do recurso

}
```

Fonte: Autoral

//OBS: OS CÓDIGOS DO PROGRAMA SE ENCONTRAM NA PASTA DO TRABALHO

REFERÊNCIAS

COULOURIS, George et al. **Sistemas Distribuídos:** Conceitos e Projetos. 5. ed. Porto Alegre: Bookman, 2013.