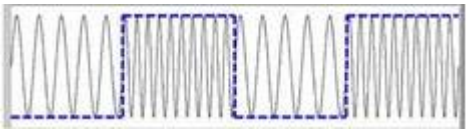


PROJETO CLIENT-SERVER EM TEMPO REAL

Neste projeto você deverá construir um código em Python para transmissão (client) e recepção (server) serial com uma resposta do server para o client. Você deverá construir seu código modificando o código anterior (loop-back)

Objetivo:

Você deverá ter duas aplicações distintas. Uma aplicação (client) deverá enviar via transmissão serial UART uma sequência de comandos (lista de bytes) para outra aplicação (server). Os comandos serão mostrados a seguir.

A sequência deve ter entre 10 e 30 comandos, a ser determinada pelo client (aleatoriamente). O server não sabe a quantidade de comandos que irá receber.

Após a recepção, o server deverá retornar ao client uma mensagem informando o número de comandos que foi recebido.

Assim que o client receber a resposta com este número, poderá verificar se todos os comandos foram recebidos, e o processo termina.

Caso o número de comandos informado pelo server não esteja correto, o cliente deverá expor uma mensagem avisando a inconsistência.

Se o server não retornar nada em até 10 segundos, o cliente deverá expor uma mensagem de “time out”

Importante! Lembre-se que o server não conhece a quantidade de comandos que serão transmitidos!

A transmissão deve ser feita com dois Arduinos. Cada aplicação irá se comunicar com um deles.

Comandos existentes:

A seguir a lista de comandos. Note que há 6 comandos diferentes. Alguns compostos por 1 byte, alguns por 2 bytes e outros compostos por 4 bytes.

Comando 1: **00 FF 00 FF** (comando de 4 bytes)

Comando 2: **00 FF FF 00** (comando de 4 bytes)

Comando 3: **FF** (comando de 1 byte)

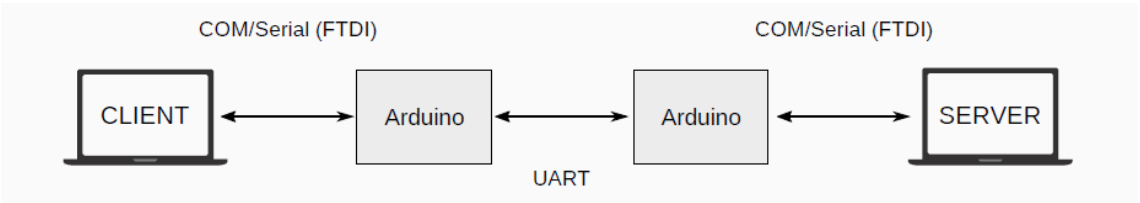
Comando 4: **00** (comando de 1 byte)

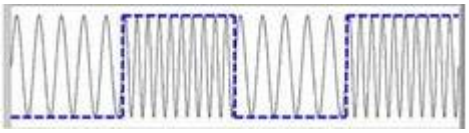
Comando 5: **FF 00** (comando de 2 bytes)

Comando 6: **00 FF** (comando de 1 bytes)

Montagem

Você terá que descobrir como conectar os Arduinos. Precisará de jumpers.





CAMADA FÍSICA DA COMPUTAÇÃO

ENGENHARIA DA COMPUTAÇÃO - Rodrigo Carareto

Gerando a sequência de comandos a serem enviados.

O client deve sortear um número **N** entre 10 e 30, que irá determinar a **quantidades de comandos a serem enviados**. Em seguida deve construir uma lista contendo os **N** comandos. Esta lista deve conter os comandos de 1 a 6 em uma sequência também aleatória, desconhecida pelo server e elaborada aleatoriamente pelo client.

Exemplo

Vamos imaginar que o cliente tenha sorteado um N igual a 24. Nesse caso deverá sortear 24 vezes um número inteiro do conjunto {1, 2, 3, 4, 5, 6}, compondo a sequência de comandos. Por exemplo: lista de comandos =[1 4 3 2 1 2 1 5 3 6 5 4 6 3 4 2 1 5 4 5 5 3 4 6].

Após enviar esses 24 comandos, o cliente espera receber o número 24 como resposta em até 10 segundos. Caso isso não ocorra, deverá exibir a mensagem “time out”.

Entrega 04/03/2022:

Você e sua dupla deverão apresentar para seu professor o código funcionando em 3 situações.

Você deverá simular um caso de sucesso de transmissão.

Um caso de erro (server recebeu os comandos com problema de interpretação). Nesse caso você pode forçar um erro com algo “hard coded” no seu código server.

Um caso de ausência de resposta por parte do server e mensagem de time out do cliente.

DATA MÁXIMA PARA Apresentação 04/03

Dicas!

Nesse projeto, você deverá implementar ações que obedecem uma ordenação temporal real. Para entender como isso pode ser feito é fundamental que você entenda como acontece o recebimento de dados no software que você recebeu no início do curso. Você precisará fazer alterações nas camadas de enlace. Tente responder as seguintes perguntas para verificar seu grau de compreensão dessas camadas:

- Quando você quer deixar sua aplicação esperando por uma resposta, que comando usaria?

- O aconteceria se, por exemplo, você usasse o comando do enlace: ***tipoEnlace.getData(int N)*** numa situação em que uma outra aplicação lhe enviou uma quantidade menor que *N* de bytes?

- Com base na resposta anterior, onde você deverá implementar o cronômetro para limitar o tempo de espera por respostas?