С

Cubo Animado

Objetivo:

Desenvolver uma aplicação que controle um cubo de leds, gerando animações visuais.

Equipe:

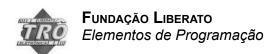
O trabalho será desenvolvido por uma equipe de 2 alunos da mesma turma.

Descrição do cubo:

O cubo de leds é uma estrutura de leds organizado em 8 camadas, cada camada com 8 linhas de 8 colunas A imagem do cubo pode ser representada por um array uint8 t 8x8x8 (512 bytes)

onde cada elemento do array representa um led do cubo, que pode ser acionado com 15 intensidades de brilho diferentes, sendo o valor 0 representando led apagado.

O sistema é capaz de armazenar 32 imagens diferentes, identificadas por um índice, que assume os valores de 0 a 31.



Protocolo de interface

O controle do cubo é feito pelo canal serial, através de um protocolo proprietário.

O canal serial está configurado para trabalhar com a seguinte configuração :

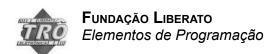
- Baudrate 115200bps
- 8 bits dados
- paridade par
- 1 stop bit

O protocolo de controle do cubo é constituído por um conjunto de comandos que apresentam, no minimo, 2 bytes de tamanho, segundo descrição abaixo :

imagem p	para o cubo CID	
CID		
512 bytes		
nao)		

Ativa o cubo CID (exibe imagem armazenada neste cubo.

Apaga o	cubo ativo
CLR	CLR



Cada comando recebido pelo cubo gera uma resposta, de 1 byte, indicando a execução do comando, conforme tabela abaixo:

ACK	Comando executado
NAK	Comando desconhecido ou invalido
MND	Pode mandar os dados
NID	indice de CID fora da faixa / inválido
TOUT	Timeout na comunicação -

CID representa o indice da imagem destino.

Valor de 0 a 31 com bit mais significativo ligado (0x80, 0x81, 0x82,0x83...)

A tabela a seguir apresenta os códigos utilizados para representação de cada comando / confirmação:

Rótulo	Valor[hex]	Valor[ascii]
SNC	0xD3	('S' 0x80)
ATX	0xD8	('X' 0x80)
CLR	0xC3	('C' 0x80)
ACK	0x41	('A')
NAK	0x4E	('N')
MND	0x4D	('M')
NID	0x49	('I')
TOUT	0x44	('T')

Tarefa

Construir uma aplicação em C++ que controle o cubo de leds, gerando uma animação visual.

Parte 1 - Modelagem

Planejar e elaborar a representação em objetos dos elementos do sistema, indicando seus atributos, métodos e inter-relações.

Descrever as classes em termos de declarações de classe em C++ ou através de diagramas de classes em UML.

Parte 2 - Implementação

Codifique a aplicação a partir da modelagem realizada, com as adequadas adaptações.

Teste a aplicação com o hardware disponibilizado.

Ferramentas UML