Circuitos digitais

Centro Universitário de Anápolis - UniEvangélica

Ana Caroliny Amancio Veiga

Lucas Galvão Lima

Mauricio Luan

Relatório de implementação de uma tabela verdade em Arduino.

Profº Me. Alexandre Tannus

Anápolis Agosto - 2018

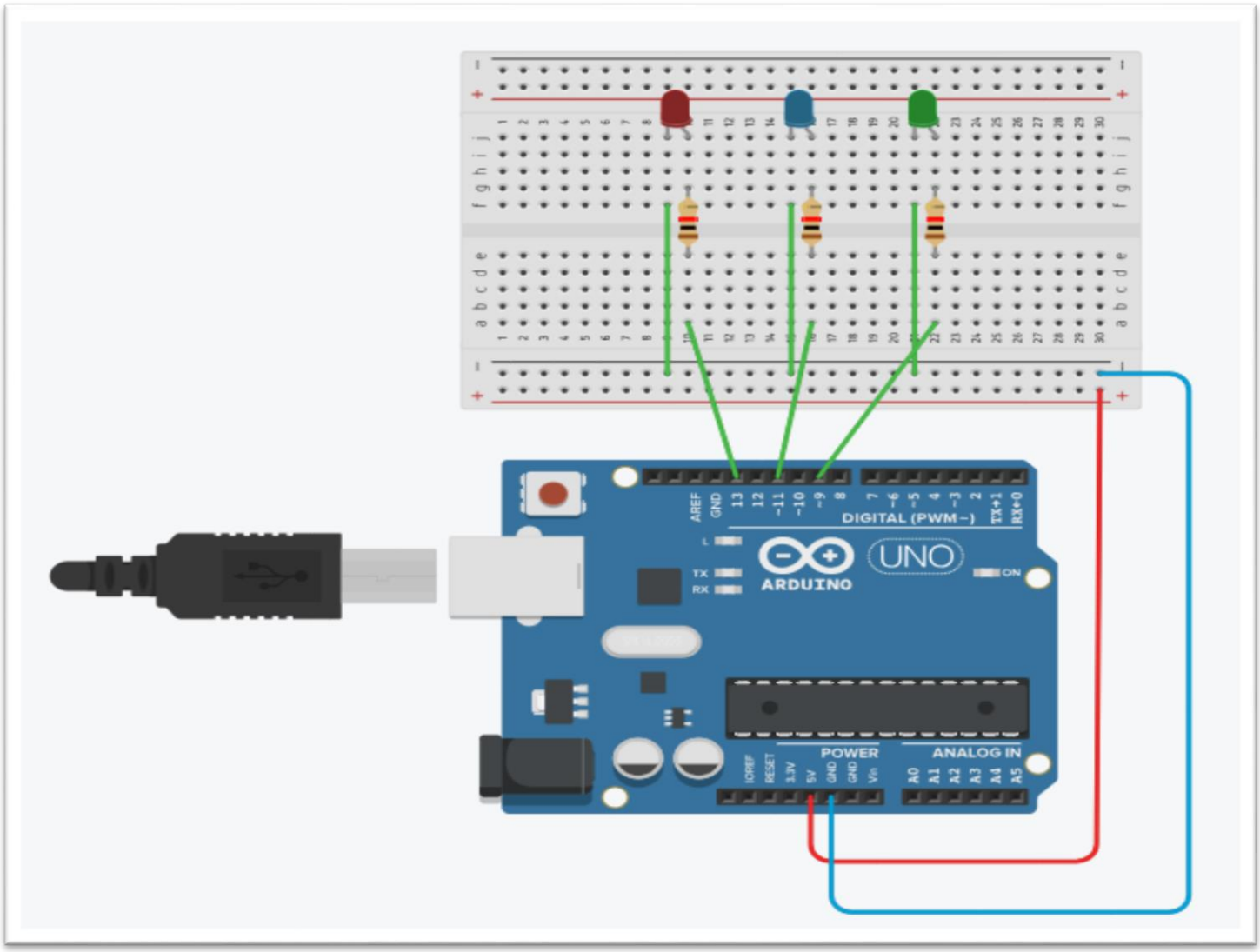
Introdução

Este relatório apresentará uma explicação breve de como foi feito e qual lógica foi para desenvolvimento do trabalho com base nos conceitos aprendidos em sala de aula e nas disciplinas de programação, redes de computadores e demais.

Materiais usados

Primeiramente foi feito uso da ferramenta online Autodesk Tinkercard® para montage do circuito por ausência de um arduíno físico.
O projeto usou:

1 placa de arduíno modelo Uno R3
1 Placa protoboard Breadboard Small
3 resistores de 10k 1/4w
3 leds 1 na cor vermelha; 1 na cor azul; 1 na cor verde
6 fios para ligar o arduíno à placa
2 fios para ligar a protoboard ao arduíno sendo 1 ligado no positivo, outro negativo.



Funcionamento…

A princípio, desenvolvemos o circuito de modo que fosse possível visualizar como funciona cada componente sem que suas linhas se traçassem. A lógica usada no código parte do princípio da estrutura da linguagem C (logo, que a linguagem do arduíno é uma linguagem C-like), usamos criação do *setup* para definir as portas a serem usadas no circuito conforme mostra o código abaixo:

```
void setup()
{
        //RED - entrada de dados(valores lógicos) na porta 13
        pinMode(13, INPUT);
        //BLUE - entrada de dados(valores lógicos) na porta 11
        pinMode(11, INPUT);
        //GREEN - resultado da comparação
        pinMode(9, OUTPUT);
}
```

Logo após, criamos as funções anteriormente à criação do método principal da maneira que está sendo representada abaixo:

```
//função and (A saída é "verdadeira" quando ambas as entradas são "verdadeiras")
int funcaoAnd(int a, int b){
    return a && b;
}

//função Or(A saída é verdadeira se uma delas forem verdadeiras - não ambas)
    int funcaoOr(int a, int b){
    return a || b;
}

//função Nand(
    int funcaoNand(int a, int b){
    return !(a && b);
}
```

```
//função de negação
	int funcaoNot(int a){
	return !a;
}

//Função Nor
int funcaoNor(int a, int b){
	return !(a || b);
}

//Função Xor
int funcaoXor(int a, int b){
	return a ^ b;
}

//Função Xnor
int funcaoXnor(int a, int b){
	return !(a ^ b);
}
```

O ultimo trecho do nosso código é o principal, na linguagem C chamamo-no de *Main.* Nesse trecho, representaremos o funcionamento do nosso código juntamente com as chamadas das funções.

```
//função principal da aplicação
void loop()
{
	//chama a função And já criada
	delay(1000);
	digitalWrite(13, HIGH);
	digitalWrite(11, LOW);
```

```
        a = digitalRead(13);
        b = digitalRead(11);
        digitalWrite(9, funcaoAnd(a,b));

        digitalWrite(13, LOW);
        digitalWrite(11, LOW);
        a = digitalRead(13);
        b = digitalRead(11);
        digitalWrite(9, funcaoOr(a,b));

        digitalWrite(13, LOW);
        digitalWrite(11, HIGH);
        a = digitalRead(13);
        b = digitalRead(11);
        digitalWrite(9, funcaoNor(a,b));

        delay(1000);
        digitalWrite(13, HIGH);
        digitalWrite(11, HIGH);
        a = digitalRead(13);
        b = digitalRead(11);
        digitalWrite(9, funcaoNot(a,b));
        delay(1000);
}
```

Lições aprendidas

O presente trabalho fez que desenvolvéssemos pensamento sistêmico sobre uso de hardware com o funcionamento de software por trás. O desenvolvimento do presente projeto fez com que corrêssemos atrás de melhorias para o projeto tanto esteticamente como no seu algoritmo que impacta no seu funcionamento.