CENTRO UNIVERSITÁRIO DE ANÁPOLIS – UNIEVANGÉLICA

CIRCUITOS DIGITAIS

Acadêmica: Cíntia da Silva Galvão

Relatório Técnico Sobre o Funcionamento do Código Em Arduíno: Portas Lógicas

Anápolis – GO

2019

```ino
int verde = 11;
int azul = 12;
int vermelho = 13;

void setup () {
  pinMode (verde, OUTPUT);
  pinMode (azul, OUTPUT);
  pinMode (vermelho, OUTPUT);
}

void loop() {
  bool valor1[4]={false, false, true, true};
  bool valor2[4]={false, true, false, true};

  for (int i=0; i<4; i++) {
    digitalWrite(azul, valor1[i]);
    digitalWrite(vermelho, valor2[i]);
    digitalWrite(verde, andPorta(valor1[i], valor2[i]));
    delay(1500);
  }

  for (int i=0; i<4; i++) {
    digitalWrite(azul, valor1[i]);
    digitalWrite(vermelho, valor2[i]);
    digitalWrite(verde, or_p(valor1[i], valor2[i]));
    delay(1500);
  }

  for (int i=0; i<4; i++) {
    digitalWrite(azul, valor1[i]);
    digitalWrite(vermelho, valor2[i]);
    digitalWrite(verde, nor_p(valor1[i], valor2[i]));
    delay(1500);
  }

  for (int i=0; i<4; i++) {
    digitalWrite(azul, valor1[i]);
    digitalWrite(vermelho, valor2[i]);
    digitalWrite(verde, xor_p(valor1[i], valor2[i]));
    delay(1500);
  }

  for (int i=0; i<4; i++) {
    digitalWrite(azul, valor1[i]);
    digitalWrite(vermelho, valor2[i]);
    digitalWrite(verde, xnor_p(valor1[i], valor2[i]));
    delay(1500);
  }
}

bool andPorta(bool x, bool y) {
  if (x && y){
    return HIGH;
  }
  else{
    return LOW;
  }
}

bool or_p(bool x, bool y) {
  if (x || y){
      return HIGH;
  }
  else{
    return LOW;
  }
}

bool nand_p(bool x, bool y) {
  if (!(x && y)){
      return HIGH;
  }
  else{
    return LOW;
  }
}

bool nor_p(bool x, bool y) {
  if (!(x || y)){
      return HIGH;
  }
  else{
    return LOW;
  }
}

bool xor_p(bool x, bool y) {
  if (!x && y || x && !y){
          return HIGH;
  }
  else{
    return LOW;
  }
}

bool xnor_p(bool x, bool y) {
  if (!(!x && y || x && !y)){
          return HIGH;
  }
  else{
    return LOW;
  }
}
```

int verde = 11; Está atribuindo o valor e o tipo da variável.

int azul = 12;

int vermelho = 13;

---

void setup () { Essa parte do código está atribuindo a "função" de cada variável

pinMode (verde, OUTPUT); ex: a variável "verde" será saída

pinMode (azul, OUTPUT);

pinMode (vermelho, OUTPUT);

}

---

void loop() {

bool valor1[4]={false, false, true, true}; Aqui mostra que terá um loop baseado nestes valores. Os valores

bool valor2[4]={false, true, false, true}; representam a tabela verdade.

---

digitalWrite(azul, valor1[i]) – Está escrevendo na porta azul, ou seja, número 12 o valor contido na posição i do vetor valor1. digitalWrite(vermelho, valor2[i]) - Está escrevendo na porta vermelho, ou seja, número 13 o valor contido na posição i do vetor valor2.

for (int i=0; i<4; i++) {

digitalWrite(azul, valor1[i]);

digitalWrite(vermelho, valor2[i]);

digitalWrite(verde, andPorta(valor1[i], valor2[i]));

delay(1500);

}

---

bool and_porta(bool x, bool y) Recebe o valor contido no vetor1[i] na variável X e recebe o valor contido no **vetor2[i] na variável Y. return x && y** Retorna o resultado da combinação entre dois valores booleanos correspondentes a AND

Recebe o valor contido no vetor1[i] na variável X e recebe o valor contido no vetor2[i] Retorna o resultado da combinação entre dois valores booleanos correspondente a OR.

bool andPorta(bool x, bool y) {

if (x && y){

return HIGH;

}

```
  else{
    return LOW;
  }
}
```

```
bool nand_p(bool x, bool y) {
  if (!(x && y)){
        return HIGH;
  }
  else{
    return LOW;
  }
}
```

bool nand_port(bool x, bool y) – Recebe o valor contido no vetor1[i] na variável Apresenta o resultado da combinação entre dois valores booleanos de acordo com a porta lógica NAND.

Recebe o valor contido no vetor1[i] na variável X e Apresenta o resultado da combinação entre dois valores booleanos de acordo com a porta lógica NOR.

```
bool nor_p(bool x, bool y) {
  if (!(x || y)){
        return HIGH;
  }
  else{
    return LOW;
  }
}
```

Recebe o valor contido no vetor1[i] na variável X e recebe o valor contido no vetor2[i] na variável Y. apresenta !x && Mostra o resultado da combinação entre dois valores booleanos de acordo com a porta lógica XOR.

```
bool xor_p(bool x, bool y) {
  if (!x && y || x && !y){
```

```
        return HIGH;
  }
  else{
    return LOW;
  }
}
```

Recebe o valor contido no vetor1[i] na variável X e recebe o valor contido no vetor2[i] na variável Y. !(!x && apresenta o resultado da combinação entre dois valores booleanos de acordo com a porta lógica XNOR.

```
bool xnor_p(bool x, bool y) {
  if (!(!x && y || x && !y)){
        return HIGH;
  }
  else{
    return LOW;
  }
}
```