CENTRO UNIVERSITÁRIO DE ANÁPOLIS - UNIEVANGÉLICA

Luís Fernando Montes
Samara de Lourdes

RELATÓRIO DE CIRCUITOS DIGITAIS

Anápolis
2019

RELATÓRIO DA IMPLEMENTAÇÃO DE PORTAS LÓGICAS NO ARDUÍNO

Código abaixo em linguagem C:

```
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, and_port(valor1[i], valor2[i]));
    delay(1500);
  }

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

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

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

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

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

bool and_port(bool x, bool y) {
  return x && y;
}

bool or_port(bool x, bool y) {
  return x || y;
}

bool nand_port(bool x, bool y) {
  return !(x && y);
}
```

```
69.
70. bool nor_port(bool x, bool y) {
71.   return !(x || y);
72. }
73.
74. bool xor_port(bool x, bool y) {
75.   return !x && y || x && !y;
76. }
77.
78. bool xnor_port(bool x, bool y) {
79.   return !(!x && y || x && !y);
80. }
```

Nas linhas 1, 2 e 3 estamos criando as variáveis referentes a cada um dos LEDs, cada variável está recebendo um valor que é número da porta onde o LED está conectado no arduíno.

- LED verde na porta 11;
- LED azul na porta 12;
- LED vermelha na porta 13.

Na linha 5 temos a função SETUP onde são realizadas as configurações iniciais que serão executadas quando o sistema for ligado ou reiniciado. Dentro dela temos a função pinMode onde é configurada uma porta como entrada ou saída, o primeiro parâmetro é o número da porta e o segundo parâmetro diz se ela é entrada (INPUT) ou saída (OUTPUT).

- pinMode (verde, OUTPUT) – Está configurando a porta verde, ou seja, número 11 como saída.
- pinMode (azul, OUTPUT) – Está configurando a porta azul, ou seja, número 12 como saída.
- pinMode (vermelho, OUTPUT) – Está configurando a porta vermelho, ou seja, número 13 como saída.

Na linha 11 temos a função LOOP onde começa um laço de execução infinito que só será parado caso o sistema seja desligado ou ocorra falha na execução.
Na linha 12 e 13 estão sendo criados dois vetores de tamanho 4 com valores booleanos, assim como numa tabela verdade. A tabela verdade desses vetores seria:

Valor1	Valor2
0	0
0	1
1	0
1	1

Após a definição dos vetores são criados 6 laços FOR, cada um para representar uma porta lógica, dentro desses laços é utilizada a variável i como contador para percorrer as 4 posições de cada um dos dois vetores, dessa forma, fazendo as combinações dos dois valores (TRUE ou FALSE) que serão mostradas pelos LEDs azul e vermelho, já o LED verde será o resultante da combinação entre os outros 2 de acordo com a porta lógica.

Dentro de cada FOR temos a função digitalWrite onde é escrito um valor na porta digital, no primeiro parâmetro temos o número da porta e o segundo parâmetro o valor de TRUE para ligado ou o valor de FALSE para desligado. Como estamos utilizando vetores com valores booleanos no segundo parâmetro da digitalWrite chamamos o vetor (valor1 ou valor2) e utilizamos a variável i para que sejam

percorridos todos os valores dele até que execute todos os valores do vetor e todo o laço FOR.

Em cada FOR teremos duas funções digitalWrite que irá se repetir em todos eles, a primeira está escrevendo na porta azul e a segunda na porta vermelha. São essas funções abaixo:

- 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.

Apenas a terceira função digitalWrite que será diferente em cada FOR pois ela varia de acordo com a porta lógica.

No final de cada laço for temos a função delay(1500), dentro dessa função colocamos em milissegundos o tempo de pausa para que seja possível ver com mais clareza o funcionamento dos LEDs.

Na linha 15 temos a digitalWrite do primeiro FOR que é referente a porta AND.

- digitalWrite(verde, and_port(valor1[i], valor2[i])) - Está escrevendo na porta verde, ou seja, número 11 o valor referente ao resultado da combinação *(valor da posição i do vetor valor1 + o valor da posição i do vetor valor2)* de acordo com a porta lógica AND.

Na linha 25 temos a digitalWrite do segundo FOR que é referente a porta OR.

- digitalWrite(verde, or_port(valor1[i], valor2[i])) - Está escrevendo na porta verde, ou seja, número 11 o valor referente ao resultado da combinação *(valor da posição i do vetor valor1 + o valor da posição i do vetor valor2)* de acordo com a porta lógica OR.

Na linha 32 temos a digitalWrite do terceiro FOR que é referente a porta NAND.

- digitalWrite(verde, nand_port(valor1[i], valor2[i])) - Está escrevendo na porta verde, ou seja, número 11 o valor referente ao resultado da combinação *(valor da posição i do vetor valor1 + o valor da posição i do vetor valor2)* de acordo com a porta lógica NAND.

Na linha 39 temos a digitalWrite do quarto FOR que é referente a porta NOR.

- digitalWrite(verde, nor_port(valor1[i], valor2[i])) - Está escrevendo na porta verde, ou seja, número 11 o valor referente ao resultado da combinação *(valor da posição i do vetor valor1 + o valor da posição i do vetor valor2)* de acordo com a porta lógica NOR.

Na linha 46 temos a digitalWrite do quinto FOR que é referente a porta XOR.

- digitalWrite(verde, xor_port(valor1[i], valor2[i])) - Está escrevendo na porta verde, ou seja, número 11 o valor referente ao resultado da combinação *(valor da posição i do vetor valor1 + o valor da posição i do vetor valor2)* de acordo com a porta lógica XOR.

Na linha 53 temos a digitalWrite do sexto FOR que é referente a porta XNOR.

- digitalWrite(verde, xnor_port(valor1[i], valor2[i])) - Está escrevendo na porta verde, ou seja, número 11 o valor referente ao resultado da combinação

(valor da posição i do vetor valor1 + o valor da posição i do vetor valor2) de acordo com a porta lógica XNOR.

Após os laços de FOR temos as funções booleanas referentes a cada uma das portas lógicas, onde é feito o tratamento necessário para que dê o resultado de acordo com a regra da porta lógica.

O valor contido no vetor1[i] de acordo com a sua posição é passado para variável booleana X, e o valor contido no vetor2[i] de acordo com a sua posição é passado para variável booleana Y, após isso a função retorna um resultado booleano de acordo com a porta lógica, esse resultado é utilizada no laço FOR dentro do terceiro digitalWrite que apresenta o resultado no LED verde.

Função booleana da porta AND:

- bool and_port(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 de acordo com a porta lógica AND.

Função booleana da porta OR:

- bool or_port(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 de acordo com a porta lógica OR.

Função booleana da porta NAND:

- bool nand_port(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 de acordo com a porta lógica NAND.

Função booleana da porta NOR:

- bool nor_port(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 de acordo com a porta lógica NOR.

Função booleana da porta XOR:

- bool xor_port(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 || x && !y – Retorna o resultado da combinação entre dois valores booleanos de acordo com a porta lógica XOR.

Função booleana da porta XNOR:

- bool xnor_port(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 || x && !y) – Retorna o resultado da combinação entre dois valores booleanos de acordo com a porta lógica XNOR.