# Instrumentação com arduino
# Função Millis() e Buffer Cirular

Gustavo Leal

12 de maio de 2018

# Sumário

# Lista de Figuras

# Capítulo 1

# Delay()

Sistemas eletrônicos muitas vezes precisam de esperas entre execuções de um mesmo componente. Estas esperas, ou retardos, são conhecidos como delays (atrasos). Como todo bom controlador, o Arduino também oferece esta função para seus sistemas.

O exemplo mais simples com o uso de um delay é o Blink, um exemplo pronto fornecido pela IDE do Arduino com a simples função de piscar um LED a cada 1 segundo. O LED liga, aguarda 1 segundo e desliga por mais 1 segundo. Com o loop padrão da função main(), o LED segue piscando indeterminadamente. Mas, se incrementarmos o sistema com um segundo LED que deverá piscar a cada 2 segundos, o primeiro passará a piscar em uma frequência menor. Isso se dá porque cada delay faz com que a execução do sistema seja interrompida pelo tempo definido. Logo, o primeiro LED funciona por um ciclo, mas antes de poder iniciar o próximo ciclo ele é impedido pelo segundo LED. Ou seja, ficará 1 segundo ligado e 5 segundos desligado.

```
1  void setup() {
2    pinMode(7, OUTPUT);
3    pinMode(3, OUTPUT);
4  }
5  void loop() {
6    digitalWrite(7, HIGH);   // Aciona saida do pino 7.
7    delay(1000);             // Espera um segundo.
8    digitalWrite(7, LOW);    // Desliga a saida do pino 7.
9    delay(1000);             // Espera um segundo.
10   digitalWrite(3, HIGH);   // Aciona saida do pino 3.
11   delay(2000);             // Espera dois segundo.
12   digitalWrite(3, LOW);    // Aciona saida do pino 3.
13   delay(2000);             // Espera dois segundo.
14 }
```

Quando lidamos com aplicações maiores, o uso da função delay() se torna problemático. Existem outras soluções para contornar este problema e manter as esperas entre uma execução e outra.

# Capítulo 2

# Millis()

A função Millis() é nativa do arduino. Ela retorna um dado do tipo unsigned long contendo o tempo decorrido desde o momento em que o arduino foi ligado. Armazenando este valor, podemos calcular o tempo decorrido entre um momento e outro. Ela soluciona os problemas que a função delay apresenta, porém também possui alguns contras. A estrutura básica para utilizar a função é esta:

```
1
2         unsigned long lastTimeLed1 = 0;
3         unsigned long lastTimeLed2 = 0;
4
5         int ledState = LOW;
6         int led2State = LOW;
7         void setup() {
8
9           pinMode(3, OUTPUT);
10          pinMode(7, OUTPUT);
11
12        }
13
14        void loop() {
15          if( millis() - lastTimeLed1 > 1000){
16            lastTimeLed1 = millis();
17
18            if(ledState == LOW){
19              ledState = HIGH;
20            } else {
21              ledState = LOW;
22            }
23            digitalWrite(3, ledState);
24          }
25
26          if( millis() - lastTimeLed2 > 2000){
27            lastTimeLed2 = millis();
28            if(led2State == LOW){
29              led2State = HIGH;
30            } else {
31              led2State = LOW;
32            }
33            digitalWrite(7, led2State);
34          }
35        }
```

Com um pouco mais de linhas de código, os dois leds passam a piscar simultâneamente. Com uma estrutura semelhante, a função millis() pode ser aplicada em diversos projeto que exijam intervalos entre execuções de partes do código.