Pro 1.0

delay() vs. millis()

RoboCode

План занятия

1. Кто считает время в Arduino
2. Забываем про delay()
3. Функция millis()

RoboCode

Кто считает время в Arduino

Мы постоянно писали delay(1000); и не задумывались откуда Arduino знает, что прошла 1с. Пришло время в этом разобраться и постичь силу таймеров.

Начнём с того, откуда вообще микроконтроллер знает, сколько проходит времени. Ведь у него нет часов! Для работы микроконтроллера жизненно важен так называемый тактовый генератор, или кварцевый генератор, или он же кварц.

Давайте найдем кварц на Arduino.

RoboCode

Кто считает время в Arduino

Кварц выполняет очень простую вещь: он пинает микроконтроллер со своей тактовой частотой, то есть 16 МГц кварц пинает МК 16 миллионов! раз в секунду. Микроконтроллер, в свою очередь зная частоту кварца, может прикинуть время между пинками (16 МГц = 0.0625 микросекунды), и таким образом ориентироваться во времени. Но на деле не всё так просто, потому что принимают пинки таймера так называемые таймеры-счётчики (Timer-counter). Это физически расположенные внутри МК устройства, которые занимаются подсчётом пинков тактового генератора. И вот микроконтроллер уже может обратиться к счётчику и спросить, а сколько там натикало. И счётчик ему расскажет. И вот этим мы уже можем пользоваться, для этого у Ардуино есть готовые функции времени.

Timer-counter

RoboCode

Любимая функция delay();

Простейшей с точки зрения использования функцией времени является задержка, их у нас две:

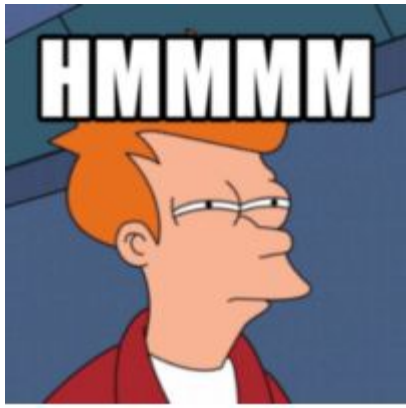
delay(time) – “Приостанавливает” выполнение кода на time миллисекунд.

delayMicroseconds(time) – Аналог delay(), приостанавливает выполнение кода на time микросекунд.

Задержки использовать очень просто:

```
6  void loop() {
7    digitalWrite(13, HIGH);
8    delay(1000);
9    digitalWrite(13, LOW);
10   delay(1000);
11 }
```

Что же делает эта программа?



RoboCode

Забываем про delay()

Так в чем же проблема? Почему использовать delay() не рекомендуется? Как вы думаете?

Мы можем делать какое-то действие раз в секунду. А что делать, если нам нужно выполнять одно действие раз в секунду, а другое – три? А третье – 10 раз в секунду, например.

Если я хочу получать значение с одного датчика каждую секунду, а с другого 10 раз в секунду?

Забываем про delay()

Основаня проблема в том, что при использовании функции delay() наша программа полностью зависает и не может никаким образом выполнять какие-то другие действия.

Поэтому рекомендуется использовать функцию millis(). Да, она сложнее в использовании, но вы уже прошли огромный путь и самое время постичь новые силы.

RoboCode

Функции millis() и micros()

Данные функции возвращают время, прошедшее с момента запуска микроконтроллера. Таких функций у нас две:

millis() – Возвращает количество миллисекунд, прошедших с запуска. Возвращает unsigned int, от 1 до 4 294 967 295 миллисекунд (~50 суток), после переполнения сбрасывается в 0.

micros() – Возвращает количество микросекунд, прошедших с запуска. Возвращает unsigned int, от 4 до 4 294 967 295 (каждые 4 микросекунды) микросекунд (~70 минут), после переполнения сбрасывается в 0.

RoboCode

Функции millis() и micros()

Вы спросите, а как время со старта МК поможет нам организовать действия по времени? Очень просто, алгоритм такой:

1) Выполнили действие
2) Запомнили текущее время со старта МК (в отдельную переменную)
3) Ищем разницу между текущим временем и запомненным
4) Как только разница больше нужного нам времени – выполняем действие

Пример:

```
int prevAction;

void setup() {
  prevAction = millis();
}

void loop() {
  if (millis() - prevAction >= 500) {    // ищем разницу
    prevActionel = millis();             // сброс таймера
    //Action                             // выполнить действие
  }
}
```

RoboCode

Задание 1:
Вывести в монитор порта время в миллисекундах, прошедшего с момента включения Arduino

А в секундах?

```
1  void setup() {
2    Serial.begin(9600);
3  }
4
5  void loop() {
6    Serial.println(millis());
7  }
```

```
1  void setup() {
2    Serial.begin(9600);
3  }
4
5  void loop() {
6    Serial.println(round(millis()/1000));
7  }
```

RoboCode

Задание 2: Сделать мигание светодиодом 2 раза в секунду, используя функцию millis()

```
1  unsigned long prevAction;
2  int ledState = 0;
3
4  void setup() {
5    pinMode(13, OUTPUT);
6  }
7
8  void loop() {
9    if (millis() - prevAction >= 500) {
10     if (ledState == 0) {
11       ledState = 1;
12     } else {
13       ledState = 0;
14     }
15     digitalWrite(13, ledState);
16     prevAction = millis();
17   }
18 }
```

```
ledState = 1 - ledState;
```

RoboCode

Задание 3:
Сделать мигание одним светодиодом 1 раз в секунду, вторым – 2 раза в секунду, третьим – 5 раз в секунду

```
unsigned long prevAction1;
unsigned long prevAction2;
unsigned long prevAction3;
int ledState1 = 0;
int ledState2 = 0;
int ledState3 = 0;

void setup() {
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11,OUTPUT);
}
```

```
void loop() {
  if (millis() - prevAction1 >= 1000) { // 1
    ledState1 = 1 - ledState1;
    digitalWrite(13, ledState1);
    prevAction1 = millis();
  }
  if (millis() - prevAction2 >= 500) {  // 2
    ledState2 = 1 - ledState2;
    digitalWrite(13, ledState2);
    prevAction2 = millis();
  }
  if (millis() - prevAction3 >= 200) {  // 5
    ledState3 = 1 - ledState3;
    digitalWrite(13, ledState1);
    prevAction3 = millis();
  }
}
```

RoboCode

Задание 4:

Сделать управление серво через потенциометр. Параллельно с этим светодиод должен мигать каждые 2 секунды

```
#include<Servo.h>

Servo servo;
unsigned long prevAction1;
int ledState1 = 0;

void setup() {
  pinMode(13, OUTPUT);
  servo.attach(5);
}

void loop() {
  if (millis() - prevAction1 >= 2000) { // 1 раз в 2 секунды
    ledState1 = 1 - ledState1;
    digitalWrite(13, ledState1);
    prevAction1 = millis();
  }
  servo.write(map(analogRead(A0), 0, 1023, 0, 180));
}
```

RoboCode