


# «Микропроцессорные средства и СИСТЕМЫ»

Лекция №6



СШФ СФУ  
П. Черёмушки  
2018 год

# «Основы программирования AVR-контроллеров на примере платы Arduino».

## АЛГОРИТМ: ПОНЯТИЕ. СВОЙСТВА . ВИДЫ . СПОСОБЫ ОПИСАНИЯ

• **Алгоритмом** называется точное и понятное предписание исполнителю совершить последовательность действий, направленных на решение поставленной задачи.

### *свойства :*

- **детерминированность** (определенность). Предполагает получение однозначного результата вычислительного процесса при заданных исходных данных;
- **Результативность**: реализуемый по заданному алгоритму вычислительный процесс должен через конечное число шагов остановиться и выдать искомый результат;
- **массовость**. Это свойство предполагает, что алгоритм должен быть пригоден для решения всех задач данного типа;
- **дискретность**. Означает деление определяемого алгоритмом вычислительного процесса на отдельные этапы (шаги), возможность выполнения которых исполнителем (компьютером) не вызывает сомнений.

# «Основы программирования AVR-контроллеров на примере платы Arduino».

## АЛГОРИТМ: ПОНЯТИЕ. СВОЙСТВА . ВИДЫ . СПОСОБЫ ОПИСАНИЯ

способы записи алгоритмов:

- словесный,
- графический,
- алгоритмический язык (программа).

- Достать ключ.
- Вставить ключ в замочную скважину.
- Повернуть ключ против часовой стрелки на 2 оборота.
- Вынуть ключ.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Arduino Web Page</title>
    <script>
      function GetSwitchAnalogData() {
        nocache = "&nocache=" + Math.random() * 1000000;
        var request = new XMLHttpRequest();
        request.onreadystatechange = function() {
          if (this.readyState == 4) {
            if (this.status == 200) {
              if (this.responseText != null) {
                document.getElementById("sw_an_data").innerHTML = this.responseText;
              }
            }
          }
        };
        request.open("GET", "ajax_switch" + nocache, true);
        request.send(null);
        setTimeout('GetSwitchAnalogData()', 1000);
      }
    </script>
  </head>
  <body onload="GetSwitchAnalogData()">
    <h1>Arduino AJAX Input</h1>
    <div id="sw_an_data">
    </div>
  </body>
</html>
```



# «Основы программирования AVR-контроллеров на примере платы Arduino».

## АЛГОРИТМ: Основные элементы блок-схемы

Название символа	Обозначение и пример заполнения	Пояснение
Процесс		Вычислительное действие или последовательность действий
Решение		Проверка условий
Модификация		Начало цикла
Предопределенный процесс		Вычисления по подпрограмме, стандартной подпрограмме
Ввод-вывод		Ввод-вывод в общем виде
Пуск-останов		Начало, конец алгоритма, вход и выход в подпрограмму
Документ		Вывод результатов на печать



# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Языки программирования: классификация и структура



# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Вид программ на языках низкого и высокого уровня

;МЕТКА	КОД	ОПЕРАНДЫ	КОММЕНТАРИЙ
			;ПРОГРАММА 1
A0	EQU	02	;УСТАНОВИТЬ A0=2
B0	EQU	03	;УСТАНОВИТЬ B0=3
	ORG	8000H	;НАЧАТЬ ПРОГРАММУ С АДРЕСА 8000H
START:	LDA	ADA0	;ЗАГРУЗИТЬ A0 В РЕГИСТР A
	MOV	B, A	;СОХРАНИТЬ A0 В РЕГИСТРЕ B
	LDA	ADB0	;ЗАГРУЗИТЬ B0 В РЕГИСТР A
	ADD	B	;ВЫЧИСЛИТЬ A0 + B0
	STA	ADSUM	;СОХРАНИТЬ РЕЗУЛЬТАТ
	HLT		;ОСТАНОВИТЬ МИКРОПРОЦЕССОР
ADA0:	DB	A0	;ЯЧЕЙКА ДЛЯ A0
ADB0:	DB	B0	;ЯЧЕЙКА ДЛЯ B0
ADSUM:	DS	1	;ЯЧЕЙКА ДЛЯ СУММЫ
	END		;КОНЕЦ ТЕКСТА ПРОГРАММЫ

```
#include <stdio.h>
#include <conio.h>
#include <windows.h>
// Пример программы, использующей для вычислений
// выражения: x = x + 1; ++x; и x++
main()
{
    char str[50];
    int x=5;
    int y=60;

    // Очистить экран<BR>
    textbackground(2);
    textcolor(4);
    clrscr();

    // Вычисление x++ и ++y
    x++;
    ++y;
    CharToOem("x=%d y=%d \n", str);
    printf(str, x, y);
    CharToOem("x=%d y=%d \n", str);
    printf(str, x++, ++y);

    // Конец программы
    CharToOem("\n Для выхода нажмите любую клавишу", str);
    printf(str);
    getch();
}
```

# «Основы программирования AVR-контроллеров на примере платы Arduino».

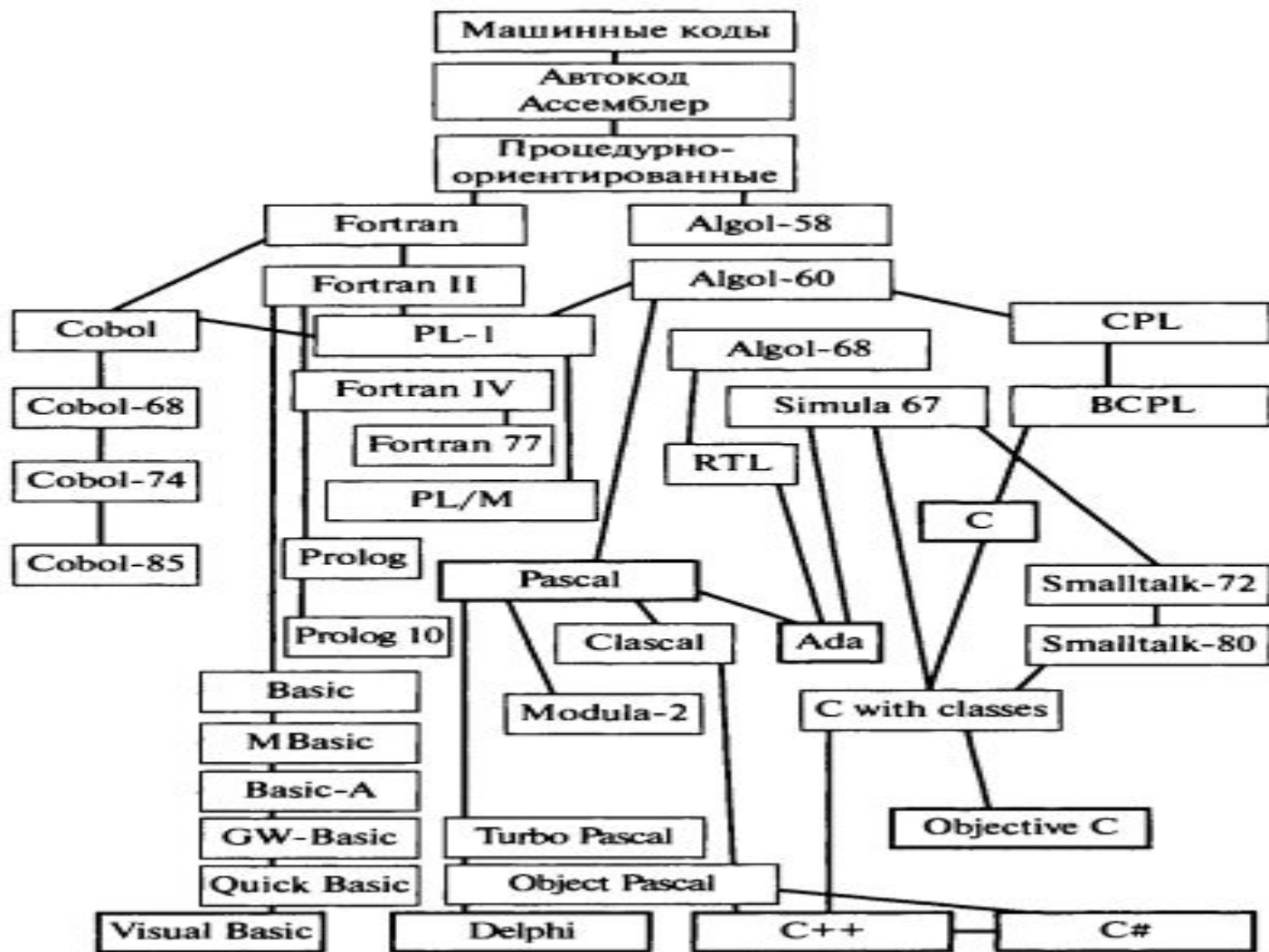
## Программирование на Arduino в машинных кодах

Байт инструкции	Что он означает для процессора
00001001	означает: взять следующий байт и запомнить его в ячейке №1
00000110	...это как раз следующий байт, который мы запоминаем в ячейке №1: число 5
00011001	означает: отнять от значения в ячейке №1 единицу и оставить там обновлённый результат
00101001	означает: сравнить значение в ячейке №1 с нулём и если оно ноль — перепрыгнуть через столько байт, сколько указано в следующем байте
00000100	...если результат был ноль, мы хотим прыгнуть через 4 байта, к предпоследней инструкции
10000011	означает, что мы хотим вывести на экран символ, код которого записан в следующем байте
01000001	...букве «А» как раз соответствует этот код
00101000	означает, что мы хотим прыгнуть назад на столько байт, сколько указано в следующем байте
00000110	...прыгать будем на 6 байт назад, к инструкции №3
10000011	означает, что мы хотим вывести на экран символ, код которого записан в следующем байте
00100001	...знаку «!» как раз соответствует этот код

**В результате исполнения такой последовательности инструкций на экран будет выведена паническая фраза «AAAA!».**

# «Основы программирования AVR-контроллеров на примере платы Arduino».

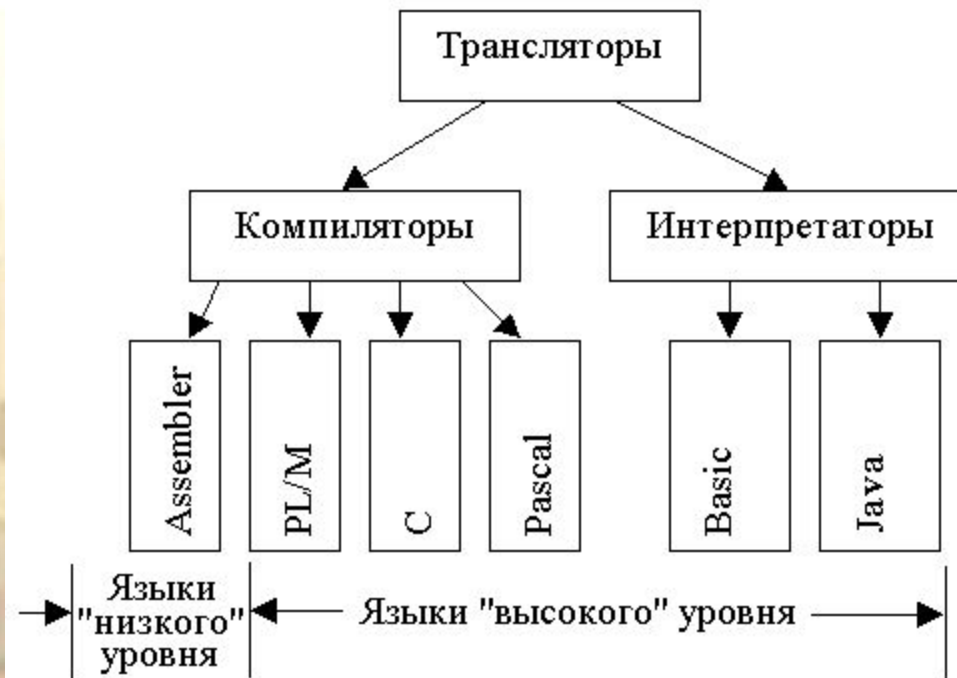
Развитие языков программирования





# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Языки программирования: трансляторы



Процесс преобразования операторов исходного языка программирования в машинные коды микропроцессора называется трансляцией исходного текста.

# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Основы программирования на Ардуино

### Основы программирования Arduino

Arduino программируется на языке C. Данный раздел ориентирован на тех, кто имеет небольшой опыт программирования, и нуждается только в пояснении особенностей языка C и интерфейса Arduino. Если все это кажется Вам немного сложным, не беспокойтесь, начинайте работать с примерами устройств и понимание придет в процессе. Для более подробного изучения основ используйте сайт [arduino.cc](http://arduino.cc).

### СТРУКТУРА

Каждая программа Arduino (часто называемая «скетч») имеет две обязательные функции (также называемые подпрограммами).

```
void setup() { }
```

Все команды, заключенные между фигурными скобками, выполняются только один раз, при первом запуске программы.

```
void loop() { }
```

Эта подпрограмма выполняется циклически вплоть до отключения питания, после завершения подпрограммы `setup()`.

### Синтаксис

Требования к форматированию в языке C вызывают некоторые затруднения у начинающих (с другой стороны, благодаря своей структуре, язык C обладает большими возможностями). Если Вы запомните следующие правила, этого будет вполне достаточно.

`//` (однострочный комментарий)  
Часто используется для размещения в тексте программы комментариев. Можно пояснить, что значит каждая строка программы. Все что размещается после двойной черты и до конца строки будет игнорироваться компилятором.

`{ }` (фигурные скобки)  
Используются для определения начала и конца блока команд (используются в функциях и циклах).

`/* */` (многострочный комментарий). Вы можете использовать эту структуру, если Вам надо создать подробный комментарий на нескольких строках. Все находящееся между этими символами будет игнорироваться компилятором.

`;` (точка с запятой)  
Каждая команда должна заканчиваться этим символом (потерянная точка с запятой — наиболее распространенная ошибка, приводящая к невозможности компиляции).

# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Основы программирования на Ардуино

### Переменные

Любая программа всего лишь определенным образом манипулирует числами. Переменные помогают жонглировать цифрами.

**int** (целочисленная)  
Основная рабочая лошадка, хранится в памяти с использованием двух байт (16 бит). Может содержать целое число в диапазоне -32 768 ... 32 767.

**long** (длинная)  
Используется в том случае, когда не хватает емкости int. Занимает в памяти 4 байта (32 бита) и имеет диапазон -2 147 483 648 ... 2 147 483 647.

**boolean** (двоичная)  
Простой тип переменной типа True/False. Занимает только один бит в памяти.

**float** (с плавающей запятой)  
Используется для вычислений с плавающей запятой. Занимает в памяти 4 байта (32 бита) и имеет диапазон -3.4028235E+38.

**char** (символ)  
Хранит один символ, используя кодировку ASCII (например «А» =65). Использует один байт памяти (8 бит). Arduino оперирует со строками как с массивами символов.

### Математические операторы

Операторы используются для преобразования чисел.

= (присвоение) делает что-то равным чему-то (например  $x=10*2$  записывает в переменную x число 20).  
% (остаток от деления). Например  $12\%10$  дает результат 2.  
+ (сложение)  
- (вычитание)  
\* (умножение)  
/ (деление)

### Операторы сравнения

Операторы, используемые для логического сравнения.

== (равно) (Например  $12==10$  не верно (FALSE),  $5==5$  верно (TRUE).)  
!= (не равно) (Например  $12!=10$  верно (TRUE),  $5!=5$  не верно (FALSE).)  
< (МЕНЬШЕ) (Например  $12<10$  не верно (FALSE),  $12<12$  не верно (FALSE),  $12<14$  верно (TRUE).)  
> (БОЛЬШЕ) (Например  $12>10$  верно (TRUE),  $12>12$  не верно (FALSE),  $12>14$  не верно (FALSE).)



# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Основы программирования на Ардуино

### Управляющие структуры

Для определения порядка выполнения команд (блоков команд) служат управляющие структуры. Здесь приведены только основные структуры. Более подробно можете ознакомиться на сайте Arduino.

```
if (условие 1) {}  
else if (условие 2) {}  
else {}
```

Если условие 1 верно (TRUE) выполняются команды в первых фигурных скобках. Если условие 1 не верно (FALSE) то проверяется условие 2. Если условие 2 верно, то выполняются команды во вторых фигурных скобках, в противном случае выполняются команды в третьих фигурных скобках.

```
for (int i=0;  
i<число повторов;  
i++) {}
```

Эта структура используется для определения цикла. Цикл повторяется заданное число раз. Переменная i может увеличиваться или уменьшаться.

### Цифровые сигналы

`digitalWrite(pin, value);`  
Если порт установлен в режим OUTPUT, в него можно записать HIGH (логическую единицу, +5В) или LOW (логический ноль, GND).

`pinMode(pin, mode);`  
Используется, чтобы определить режим работы соответствующего порта. Вы можете использовать адреса портов 0...19 (номера с 14 по 19 используются для описания аналоговых портов 0...5). Режим может быть или INPUT (вход) или OUTPUT (выход).

`digitalRead(pin);`  
Если порт установлен в режим INPUT эта команда возвращает значение сигнала на входе HIGH или LOW.

### Аналоговые сигналы

Arduino - цифровое устройство, но может работать и с аналоговыми сигналами при помощи следующих двух команд:

`analogWrite(pin, value);`  
Некоторые порты Arduino (3,5,6,9,10,11) поддерживают режим ШИМ (широтно-импульсной модуляции). В этом режиме в порт посылаются логические единицы и нули с очень большой скоростью. Таким образом среднее напряжение зависит от баланса между количеством единиц и нулей и может изменяться в пределах от 0 (0В) до 255 (+5В).

`analogRead(pin);`

Если аналоговый порт настроен в режим INPUT, то можно измерить напряжение на нем. Может принимать значения от 0 (0В) до 1024 (+5В).



# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Справочник языка Ардуино: операторы

### Операторы

- [setup\(\)](#)
- [loop\(\)](#)

### Управляющие операторы

- [if](#)
- [if...else](#)
- [for](#)
- [switch case](#)
- [while](#)
- [do... while](#)
- [break](#)
- [continue](#)
- [return](#)
- [goto](#)

### Синтаксис

- [;](#) (semicolon)
- [{ }](#) (curly braces)
- [//](#) (single line comment)
- [/\\* \\*/](#) (multi-line comment)

<http://arduino.ru/Reference>

### Арифметические операторы

- [=](#) (assignment)
- [+](#) (addition)
- [-](#) (subtraction)
- [\\*](#) (multiplication)
- [/](#) (division)
- [%](#) (modulo)

### Операторы сравнения

- [==](#) (equal to)
- [!=](#) (not equal to)
- [<](#) (less than)
- [>](#) (greater than)
- [<=](#) (less than or equal to)
- [>=](#) (greater than or equal to)

### Логические операторы

- [&&](#) (И)
- [||](#) (ИЛИ)
- [!](#) (Отрицание)

### Унарные операторы

- [++](#) (increment)
- [--](#) (decrement)
- [+=](#) (compound addition)
- [-=](#) (compound subtraction)
- [\\*=](#) (compound multiplication)
- [/=](#) (compound division)

# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Справочник языка Ардуино: данные

### Данные

#### Константы

- [HIGH](#) | [LOW](#)
- [INPUT](#) | [OUTPUT](#)
- [true](#) | [false](#)
- [Целочисленные константы](#)
- [Константы с плавающей запятой](#)

#### Типы данных

- [boolean](#)
- [char](#)
- [byte](#)
- [int](#)
- [unsigned int](#)
- [word](#)
- [long](#)
- [unsigned long](#)
- [float](#)
- [double](#)
- [string](#) - массив символов
- [String](#) - объект класса
- [массив \(array\)](#)
- [void](#)

<http://arduino.ru/Reference>

#### Преобразование типов данных

- [char\(\)](#)
- [byte\(\)](#)
- [int\(\)](#)
- [long\(\)](#)
- [float\(\)](#)

#### Область видимости переменных и квалификаторы

- [Область видимости](#)
- [static](#)
- [volatile](#)
- [const](#)

# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Справочник языка Ардуино: функции

### Функции

<http://arduino.ru/Reference>

#### Цифровой ввод/вывод

- [pinMode\(\)](#)
- [digitalWrite\(\)](#)
- [digitalRead\(\)](#)

#### Аналоговый ввод/вывод

- [analogRead\(\)](#)
- [analogReference\(\)](#)
- [analogWrite\(\)](#)

#### Дополнительные функции ввода/вывода

- [tone\(\)](#)
- [noTone\(\)](#)
- [shiftOut\(\)](#)
- [pulseIn\(\)](#)

#### Работа со временем

- [millis\(\)](#)
- [micros\(\)](#)
- [delay\(\)](#)
- [delayMicroseconds\(\)](#)

### Математические функции

- [min\(\)](#)
- [max\(\)](#)
- [abs\(\)](#)
- [constrain\(\)](#)
- [map\(\)](#)
- [pow\(\)](#)
- [sq\(\)](#)
- [sqrt\(\)](#)

### Тригонометрические функции

- [sin\(\)](#)
- [cos\(\)](#)
- [tan\(\)](#)

### Генераторы случайных значений

- [randomSeed\(\)](#)
- [random\(\)](#)

### Внешние прерывания

- [attachInterrupt\(\)](#)
- [detachInterrupt\(\)](#)

### Функции передачи данных

- [Serial](#)


# «Основы программирования AVR-контроллеров на примере платы Arduino».

## Литература:

1. <http://arduino.cc> Официальный сайт проекта Ардуино
2. <http://arduino.ru/Reference>
3. <http://wiki.amperka.ru>
4. <https://cdn.solarbotics.com/products/documentation/ardx-eg-sola-russian.pdf>
5. <http://www.yaklass.ru/materiali?chtid=474&mode=cht>
6. [http://cs.petrSU.ru/studies/filatova\\_information/CMD\\_1996566\\_M/my\\_files/Inform/Algorithm/al-3.htm](http://cs.petrSU.ru/studies/filatova_information/CMD_1996566_M/my_files/Inform/Algorithm/al-3.htm)
7. <https://progstudy.ru/index.php/sm/article/vybor-yazyka-programmirovaniya>
8. <http://kufas.ru/programming14.htm>
9. [http://lawbooks.news/informatika\\_961/mashinnonezavisimyie-yazyiki-programmirovaniya-62597.html](http://lawbooks.news/informatika_961/mashinnonezavisimyie-yazyiki-programmirovaniya-62597.html)
10. <http://www.kniga.ru/book/4011374>



**Благодарю за внимание, готов ответить  
на Ваши вопросы**



**А.М. Волошин  
Ст. преподаватель  
СШФ СФУ**