Referência Rápida para Programação de Arduino

Professor: Ricardo Kerschbaumer

Maiores informações sobre o assunto podem ser encontradas em:
https://www.arduino.cc/en/Reference/HomePage?from=Reference.Extended

Estrutura do Programa

```
void setup() {
  // Código executado uma vez:
}

void loop() {
  // Código executado repetidamente
}
```

Estruturas de Controle

```
if (condição) {
  //declaração;
}

while(condição) {
  // declaração;
}

if (condição) {
  // declaração A
}
else {
  // declaração B
}

for (inicialização; condição; incremento)
{
  // declaração;
}
```

Detalhes de Sintaxe

; - Final de declaração
{ } - Bloco de código
[ ] - Declaração de vetores e matrizes
// - Inicia comentário de 1 linha
/* */ - Comentário de várias linhas
#define - Definição de constantes
#include - Inclusão de biblioteca

Sequencias de Caracteres

```
char Str1[15];
char Str2[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o'};
char Str3[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o', '\0'};
char Str4[ ] = "arduino";
char Str5[8] = "arduino";
char Str6[15] = "arduino";
```

Tipos de Dados (AVR MEGA)

void - utilizado em sub-rotinas
boolean – valores true ou false
Char - caractér (8b)
unsigned char (8b) de 0 a 255
byte (8b) de 0 a 255
int (16b) -32768 a 32767
unsigned int (16b) 0 a 65535
word (16b) 0 a 65535
long (32b) -2147483648 a 2147483647
unsigned long (32b) 0 to 4294967295
short (16b) -32768 a 32767
Float (32b) 3,4028235E+38 -3,4028235E+38
Double igual a float

Núm. Aleatórios

randomSeed()
random()

Ponteiros

* Operador ponteiro
& Operador referência

Matemática

min() - Mínino
max() - Máximo
abs() - Absoluto
constrain() - Limitador
map() - Mapeamento
pow() - potência
sqrt() - Raiz quadrada
sin() - Seno
cos() - Cosseno
tan() - Tangênte
log() - Logaritmo
log10() - Logaritmo (10)

Entradas/Saídas Digitais

```
pinMode() // Modo
digitalWrite() // Saída
digitalRead() // Entrada
```

Temporização

millis()
micros()
delay()
delayMicroseconds()

Operadores de Comparação

== Igual
!= Diferente
< Menor
> Maior
<= Menor ou igual
>= Maior ou igual

Saídas Avançadas

tone()
noTone()
shiftOut()
shiftIn()
pulseIn()

Interrupções

attachInterrupt()
detachInterrupt()
interrupts()
noInterrupts()
digitalPinToInterrupt()
Ex: attachInterrupt(pin, ISR, mode);

Bits e Bytes

lowByte()
highByte()
bitRead()
bitWrite()
bitSet()
bitClear()

Operadores Aritméticos

= Atribuição (X = a + 1;)
+ Soma (Y = X + 3;)
- Subtração (Z = X – Y;)
* Multiplicação (J = W * 2;)
/ Divisão (T = R / 4;)
% Módulo (M = X % 10;)

Comunicação Serial (exemplo)

```
byte byteRead;

void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    byteRead = Serial.read();
    Serial.write(byteRead);
  }
}
```

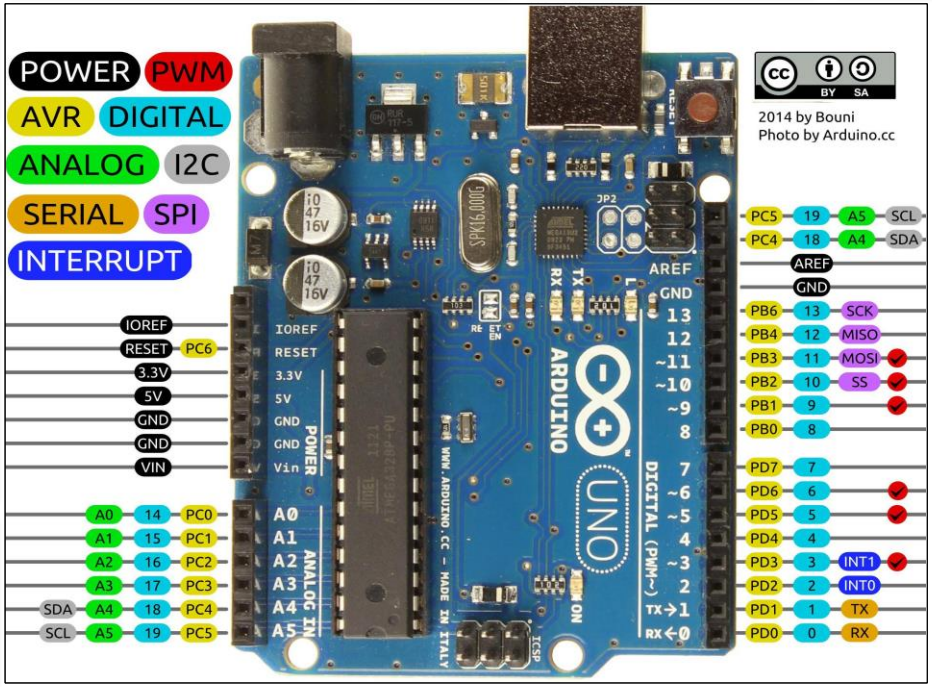
Servo Motor

```
attach()
write()
writeMicroseconds()
read()
attached()
detach()

// Exemplo de utilização
Servo myservo;
int potpin = 0;
int val;

void setup() {
  myservo.attach(9);
}

void loop() {
  val = analogRead(potpin);
  val = map(val, 0, 1023, 0, 180);
  myservo.write(val);
  delay(15);
}
```



Serial por Software

```
SoftwareSerial()
available()
begin()
isListening()
overflow()
peek()
read()
print()
println()
listen()
write()
```

Matriz

Int nomeMat[3][3];

Comunicação I2C

```
begin()
requestFrom()
beginTransmission()
endTransmission()
write()
available()
read()
SetClock()
onReceive()
onRequest()
```

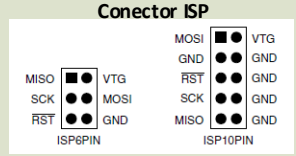
Vetores

```
int myInts[6];
int myPins[] = {2, 4, 8, 3, 6};
int mySensVals[6] = {2, 4, -8, 3, 2};
char message[6] = "hello";
```

Memória não Volátil

```
#include <EEPROM.h>
byte read(endere,o)
Write(endereco, dado)
```

Conector ISP



Entradas e Saídas Analógicas

```
analogReference()
analogRead()
analogWrite() // PWM
```

Constantes

HIGH | LOW
INPUT | OUTPUT | INPUT_PULLUP
LED_BUILTIN
true | false

Conversão

```
char()
byte()
int()
word()
long()
float()
```

Operadores Bit a Bit

& E bit a bit
| OU bit a bit
^ OU Exclusivo bit a bit
~ Negação bit a bit
<< Deslocamento a esquerda
>> Deslocamento a direita

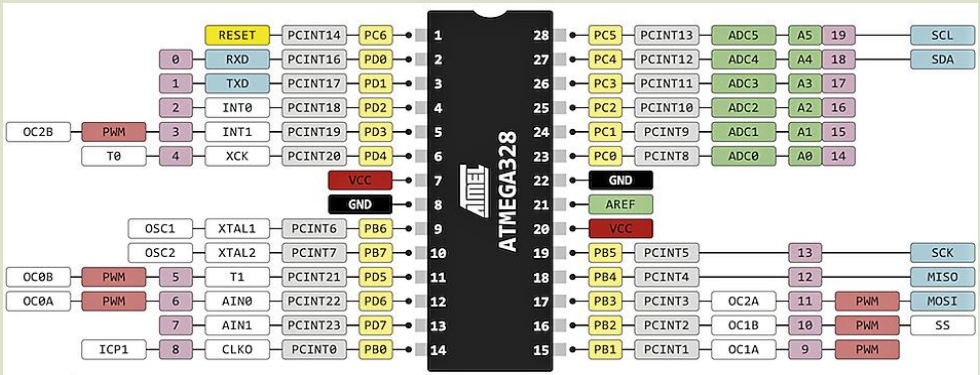
Operadores Booleanos

&& Operador E
|| Operador OU
! Operador Negado

Operadores Compostos

++ Incremento
-- Decremento
+= Adição composta
-= Subtração composta
*= Multiplicação composta
/= Divisão composta
%= Módulo composto
&= E bit a bit composto
|= OU bit a bit composto
^= XOR bit a bit composto

Microcontrolador Atmega328



Fonte: https://www.arduino.cc/en/Reference/HomePage?from=Reference.Extended