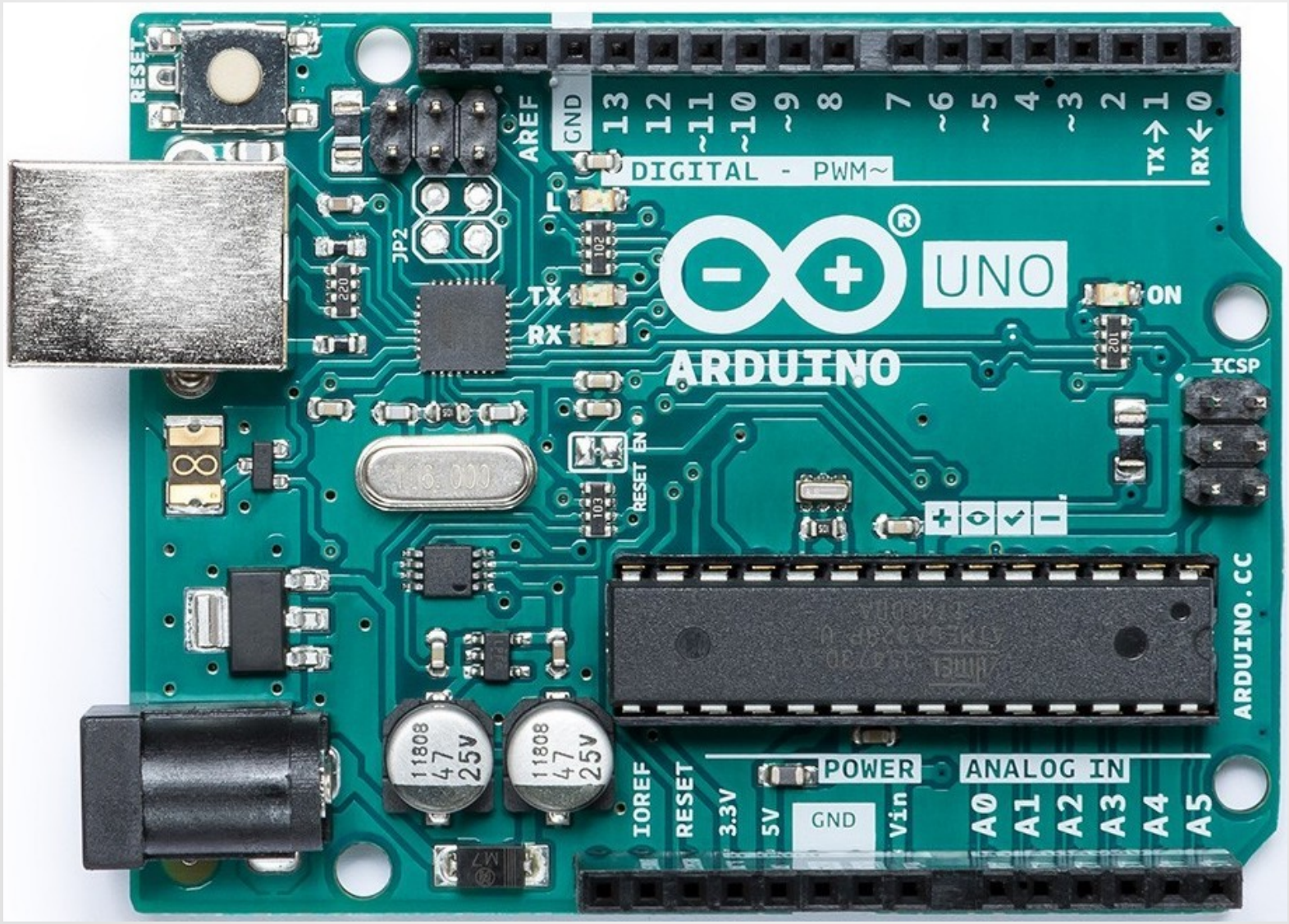
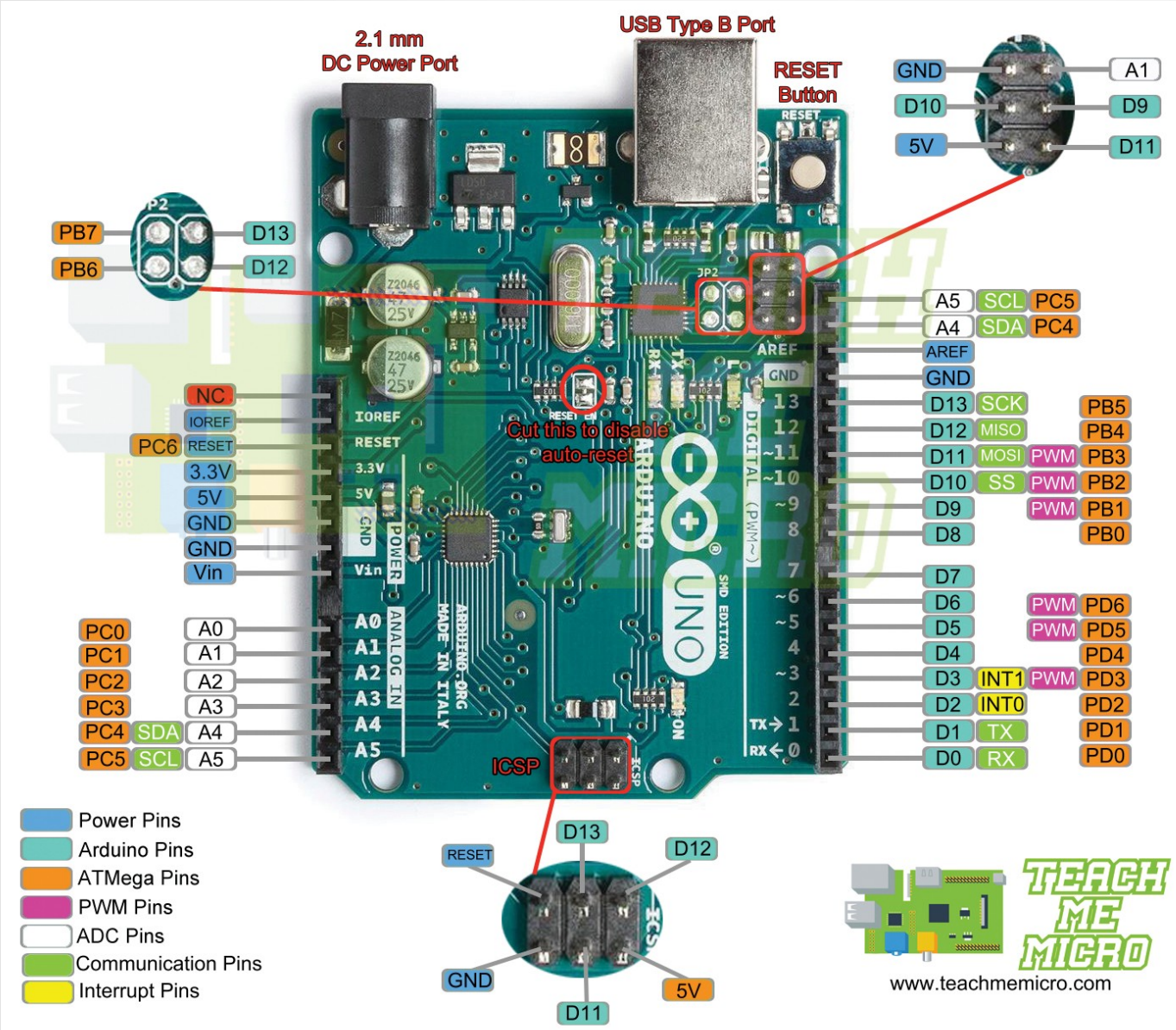
Comunicación serie Arduino/PC



Víctor . Álvarez 201911

Introdución – pinout Arduino



E/S

Dixitais:

- Pins 0 a 13
  0 a 5V, 20 mA
  Low: 0 a 2V
  High: 3 a 5V

Entradas Analóxicas

- Pins A0 – A5
  0 a 5V, 20 mA prec 1024
  0 a 3V, 50 mA prec 1024

Saídas PWM

- ~ Pins 3, 5, 6, 9, 10, 11
  0 a 5 V, 20 mA prec 256

Comunic. Serie TX/RX

- Pins 0 e 1

ICSP

- 6 pins para comunicarse directamente co proc. Atmega328
- 6 pins para programar o USB

Comunicación serie Arduino/PC

- Un porto serie é un canal de comunicación entre dous ordenadores ou dispositivos físicos. Hai moitos tipos de comunicacions serie, pero todas elas teñen en común o uso de polo menos dous conectores para a comunicacion de datos.
- No Arduino UNO estes pins son o 0 (RX, recepción) e o 1 (TX, transmisión). Polo tanto nos os podemos usar simultaneamente como E/S dixitais.
- Estes pins están fisicamente unidos ao conector USB e podémolos usar para emitir información desde o script ou comunicarnos co script empregando un terminal de texto.
- Xa imprimimos por saída serie nalgúns dos scripts anteriores. Así que para habilitar a comunicacion serie temos que ter aberta a fiestra correspondente ao ‘Monitor serie’

Víctor . Álvarez 201911

Comunicación serie Arduino/PC

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

void loop() {
  Serial.print("Mensaxe nunha linha");
  Serial.println("\tmensaxe con retorno de carro");
  Serial.print("Podemos poñer o retorno de carro ");
  Serial.print("manualmente con '\n'\n");
}
```

- Para abrir unha comunicacion co monitor serie, temos primeiro que declarala no setup(), xunto coa velocidade de comunicación en baudios (normalmente 9600).
- A continuación podemos empregar os métodos print() e println() segundo queramos mensaxes nunha liña ou mensaxes que incorporen o retorno de carro.
- Tamén se poden usar caracteres con significado especial dentro do texto, como ‘\t’ (tabulador) ou ‘\n’ (retorno de carro).

Víctor . Álvarez 201911

Comunicación serie Arduino/PC

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

void loop() {
  String mensaxe = "";
  mensaxe = "Mensaxe nunha linha";
  mensaxe += "\tmensaxe con retorno de carro";
  Serial.println(mensaxe);
  mensaxe = "Podemos ponher o retorno de carro ";
  mensaxe += "manualmente con '\n'\n";
  Serial.print(mensaxe);
}
```

- Para escribir mensaxes pola saída serie, podemos empregar variables declaradas como char ou arrays de char. Igual que en C.
- Tamén podemos empregar a potencia da POO para e tirar proveito da clase String.
- Toda a documentación da clase String está na ligazón a seguir:

  https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/
- Os métodos que proporciona esta clase facilitan moito a programación. En particular operadores como '+' son case imprescindibles.

Comunicación serie Arduino/PC

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

void loop() {
  if(Serial.available()) {
    String mensaxe = Serial.readStringUntil('\n');
    //Usamos a continuación a variable 'mensaxe'
  }
}
```

- Igual que un script pode escribir na saída serie, nós podémonos comunicar con el enviando cadeas de caracteres (arrays de char) ou mellor aínda texto que se pode tratar coa clase String.
- Para iso hai que comprobar que na ‘entrada’ serie hai datos dispoñibles, cargalos nunha variable String que posteriormente trataremos.
- Convén ter moi presente a URL da documentación da plataforma Arduino sobre a Clase Serial e os métodos que proporciona.

https://www.arduino.cc/reference/en/language/functions/communication/serial/

Comunicación serie Arduino/PC

- Revisa os scripts xa realizados, para observar como fixemos a impresión de mensaxes coa clase Serial en combinación coa clase String.
- Intenta interactuar con algúns dos scripts xa feitos, para modificar o seu comportamento.
- P.ex. no script do sensor de ultrasóns HC-SR04, elimina a entrada correspondente ao potenciómetro e tenta sustuílo por numéricos introducidos por teclado. Lembra que a clase String trata con texto, polo que habería que convertelos a valores numéricos coas funcions atoi() ou atof(): ascii to int e ascii to float, respectivamente.
- Outro exemplo máis sinxelo: modifica o brillo dun LED introducindo valores numéricos por teclado.

Comunicación serie Arduino/PC

- Nesta unidade fixemos un repaso teórico a:
  - O uso da clase Serial para habilitar e xestionar a comunicacion serie con outros dispositivos,
  - referímonos á documentación oficial para acostrumbrarnos a buscar funcións que necesitemos.
  - O uso da clase String como facilitadora do tratamento de variables que almacenen texto.
  - Non vimos nada sobre os tipos char ou array de char de C.
  - Existe a posibilidade de enviar texto ao script e que este o tome en consideración para a súa execución.
  - A entrada de datos co monitor serie sempre se interpretará como texto, polo que se queremos introducir valores numéricos temos que empregar as función atoi() ou atof().

Víctor . Álvarez 201911