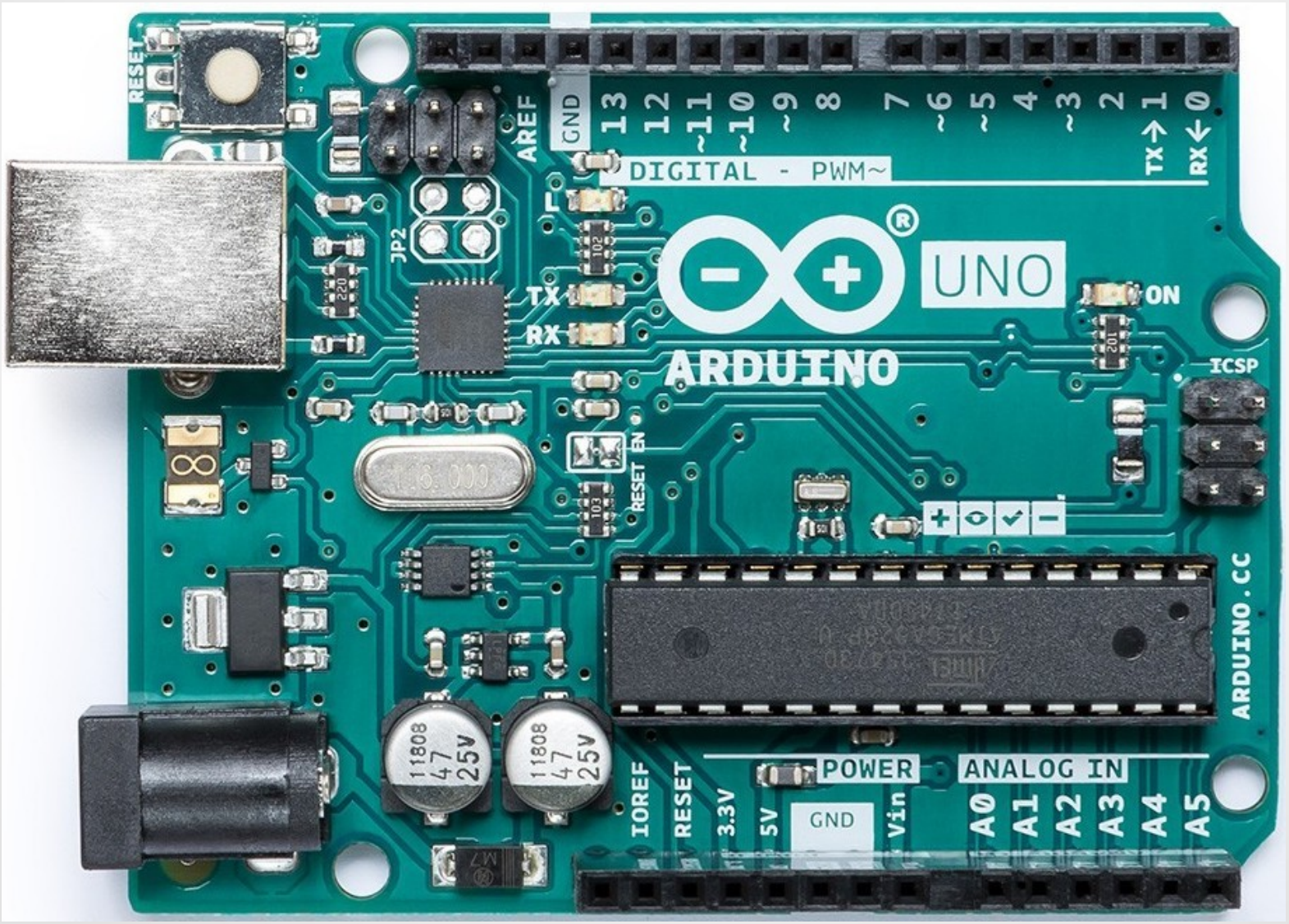
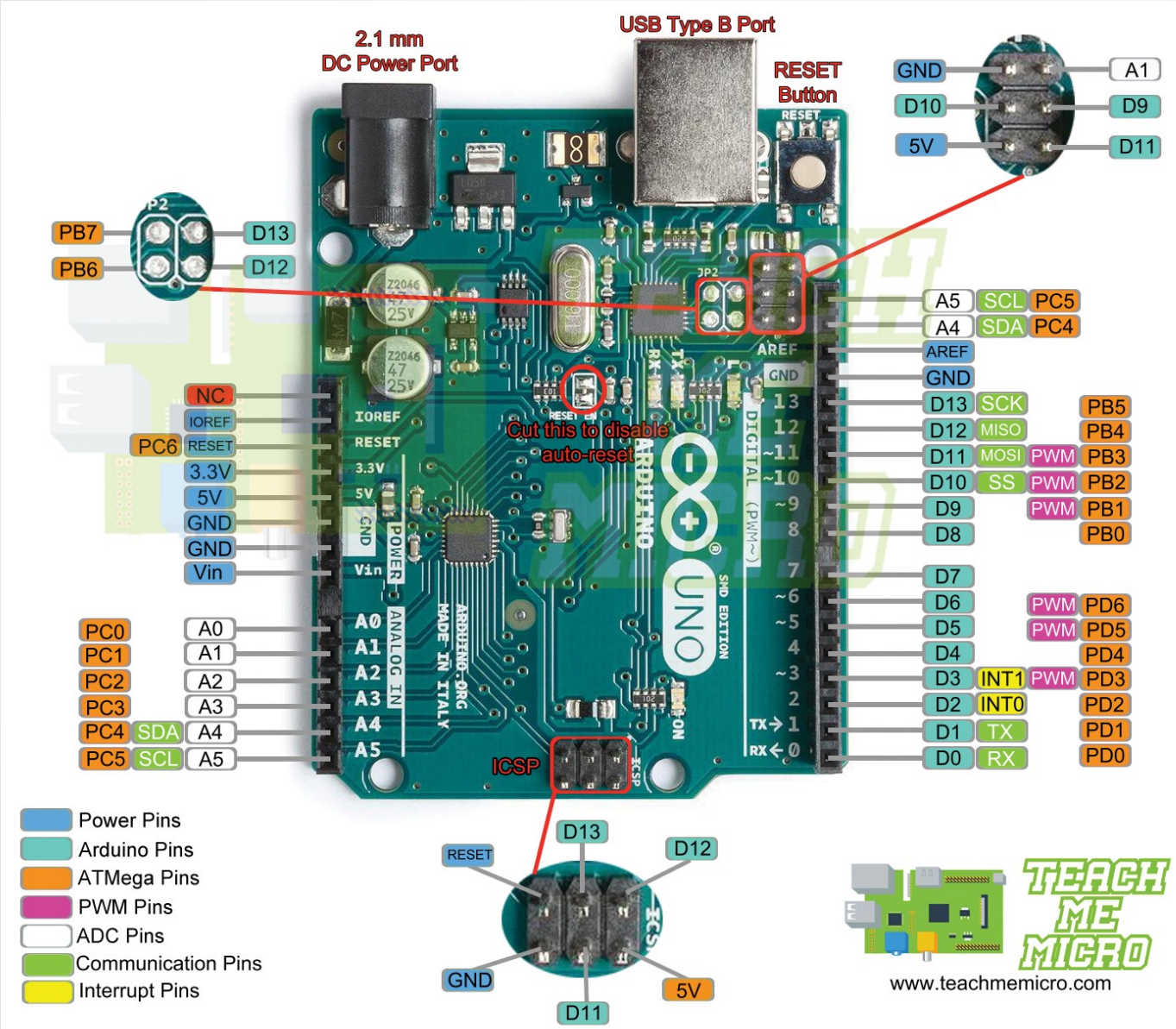
Motores paso a paso - PaP



Víctor . Álvarez 201911

Motores PaP – 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

Motores PaP

- Os motores paso a paso (PaP ou steppers, en inglés) permiten controlar o movemento cunha precision moi alta, xa que dividen o percorrido de cada volta nunha certa cantidade de pasos que se poden controlar un a un. Outra característica é que son moi estables e permanecen no seu lugar. Para vencer o seu estado é preciso superar unha resistencia mecánica alta.
- Pola súa precisión, son moi empregados en robótica, aínda que non son baratos. Nomeadamente os máis grandes. Sen embargo, para pequenos proxectos non son excesivamente caros e incluso se poden recuperar de impresoras desbotadas.
- No noso caso imos empregar o PaP 28BYJ-48, que se pode encontrar en moitos kits de Arduino ou compralos por separado a un prezo módico. Para controlalo, precisamos do módulo UNL2003, aínda que tamén é posible empregar transistores (ponte H) ou outros drivers como o L298N.

Víctor . Álvarez 201911

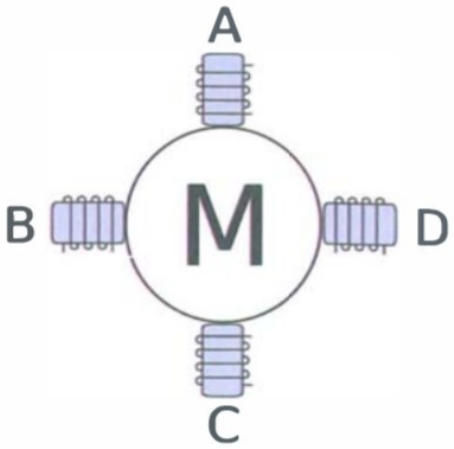
Motores PaP – stepper 28BYJ-48

- O 28BYJ-48 é un motor PaP con reductora, polo que só é válido para pequenos proxectos. Aínda así é útil para aprender a controlar outros steppers.
- Este motor pode realizar 64 pasos por volta, ademais a reductora interna ten unha relación de case 1/64. Combinadas ambas dannos un total de 4076 pasos (4096 teóricos) por volta ou o que é o mesmo 0.088º por paso, que é unha precisión moi elevada.
- A alimentación pode ser a 5V ou a 12V, segundo modelo. É preciso fixarse na etiqueta do stepper para sabelo.
- O torque ou par está ao redor do 34 Nm.
- Os consumos dependen da resistencia interna do modelo:
  - 5V: 60 Ω cun consumo de 83 mA.
  - 12V: entre 130 e 380 Ω, con consumos entre 71 e 32 mA
- Normalmente contrólase co driver ULN2003, aínda que hai outras posibilidades.

Víctor . Álvarez 201911

Motores PaP – stepper 28BYJ-48

- O 28BYJ-48 é un motor PaP unipolar de catro fases. A estructura interna das bobinas aparece na figura adxunta.
- Para mover o motor é preciso ir polarizando as bobinas de xeito secuencial. Isto marcará o sentido do xiro, a velocidade e máis o consumo e par do motor.
- Para movelo podemos polarizar unha bobina individualmente ou ben ter dúas traballando á vez. Isto permite mover o motor en pasos completos (bobina única) ou en medios pasos (dúas bobinas). Combinando ambos métodos, podemos duplicar o número de pasos por volta e dispor de máis pasos por volta.
- O método combinado é o que empregaremos para mover o 28BYJ-48. Posteriormente empregaremos a librería <Stepper.h> de Arduino, para facilitar a tarefa.

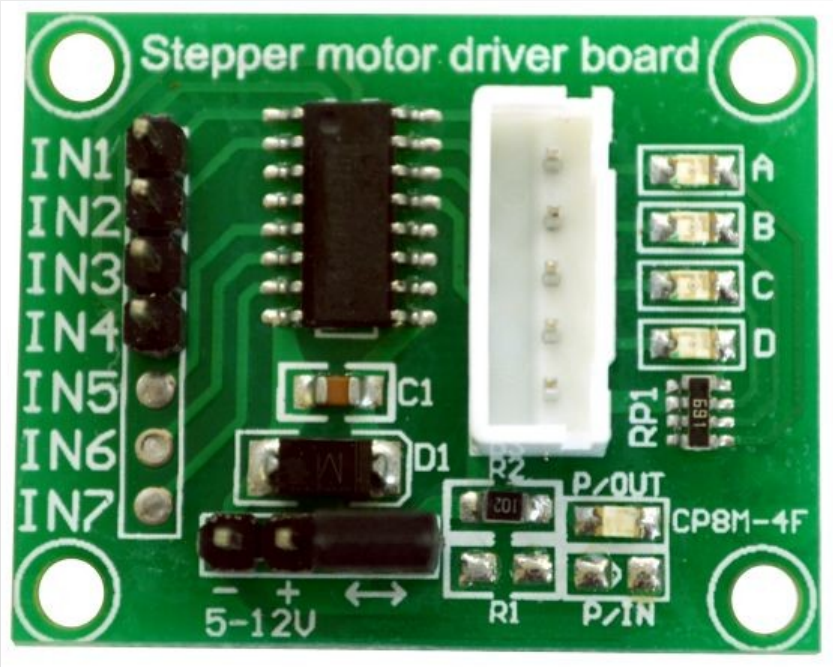


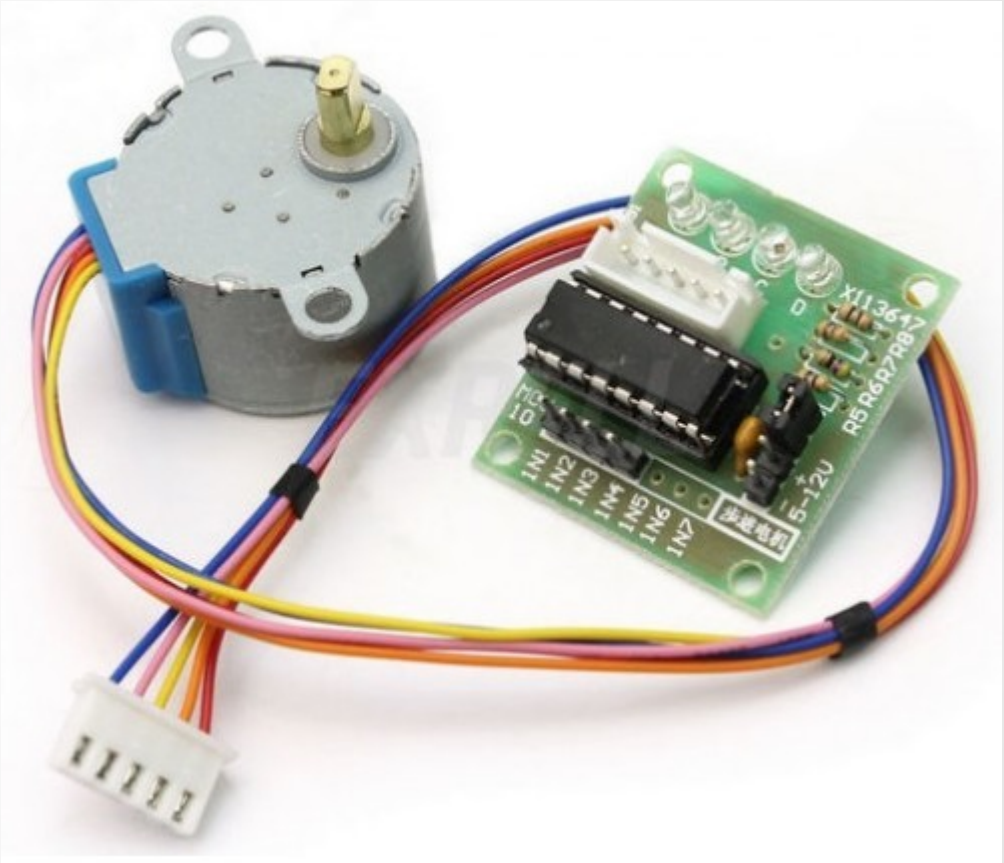
	A	B	C	D
Paso 1	ON	OFF	OFF	OFF
Paso 2	ON	ON	OFF	OFF
Paso 3	OFF	ON	OFF	OFF
Paso 4	OFF	ON	ON	OFF
Paso 5	OFF	OFF	ON	OFF
Paso 6	OFF	OFF	ON	ON
Paso 7	OFF	OFF	OFF	ON
Paso 8	ON	OFF	OFF	ON

Víctor . Álvarez 201911

Motores PaP – stepper 28BYJ-48

- O conexionado do 28BYJ-48 faise a través do driver ULN2003, coa clema de conexion incorporada. Só ten unha posición.
- Os pins IN1 a IN4 van ao Arduino e controlan o stepper.
- A alimentación pode ser entre 5V e 12 V, nos pins inferiores. Asegúrate que GND é común co Arduino.





Víctor . Álvarez 201911

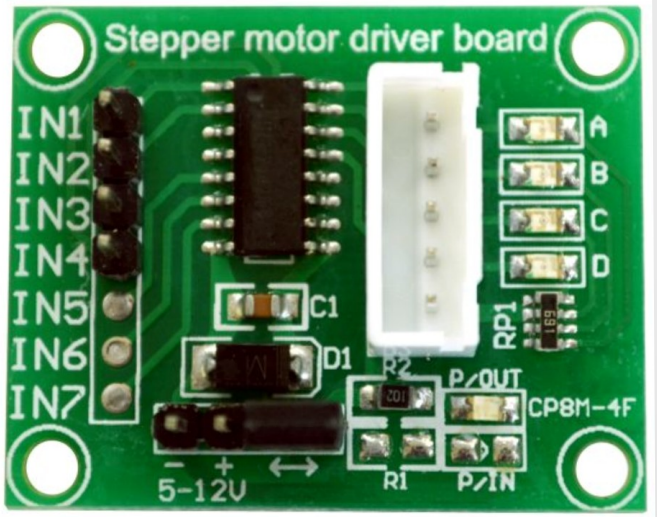
Motores PaP – stepper 28BYJ-48

motor.pap

```
/*
 * Proba o funcionamento dun motor paso a paso
 * de maneira 'manual', facendo que dea cinco
 * voltas completas en sentido horario e outras
 * cinco en sentido antihorario.
 */

//Pins para control do motor PaP
#define IN1 8
#define IN2 9
#define IN3 10
#define IN4 11

//Parámetros do motor, pasos totais e por fase
#define STEPS 4076 //Pasos totais por volta
#define NSTEPS 8 //Número de pasos por fase (4 en 1 e 2-fase)

//Conta do número de pasos realizado e intevalo temporal
int veloc = 1200;         //Velocidade do motor en us
int contador = 0;         //Leva a conta dos pasos do stepper

//Secuencias de encendido para usar co stepper
//(a) Secuencia 1-fase
//#define NSTEPS 4
//int secPasos[NSTEPS] = {B1000, B0100, B0010, B0001};

//(b) Secuencia 2-fases
//#define NSTEPS 4
//int secPasos[NSTEPS] = {B1100, B0110, B0011, B1001};

//(c) Secuencia media fase
#define NSTEPS 8
int secuenciaPasos[NSTEPS] = {B1000, B1100, B0100, B0110,
                              B0010, B0011, B0001, B1001};

void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}
```

motor.pap

```
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}

void loop() {
  //Ponhemos a andar 5 voltas en sentido horario
  for(int i = 0; i < STEPS * 5; i++) {
    moverHorario();
    delayMicroseconds(veloc);
  }
  //Ponhemos a andar 5 voltas en sentido antihorario
  for(int i = 0; i < STEPS * 5; i++) {
    moverAntiHorario();
    delayMicroseconds(veloc);
  }
}

//Funcións auxialiares para:
//(a) Movemento horario do PaP
void moverHorario() {
  contador++;
  contador %= NSTEPS;
  escribirPins(contador);
}

//(b) Movemento antihorario do PaP
void moverAntiHorario() {
  contador--;
  contador = (contador + NSTEPS) % NSTEPS;
  escribirPins(contador);
}

//(c) función auxiliar de (a) e (b), para
//    efectuar a escritura nos pins do ULN2003
void escribirPins(int paso) {
  digitalWrite(IN1, bitRead(secuenciaPasos[paso], 0));
  digitalWrite(IN2, bitRead(secuenciaPasos[paso], 1));
  digitalWrite(IN3, bitRead(secuenciaPasos[paso], 2));
  digitalWrite(IN4, bitRead(secuenciaPasos[paso], 3));
}
```



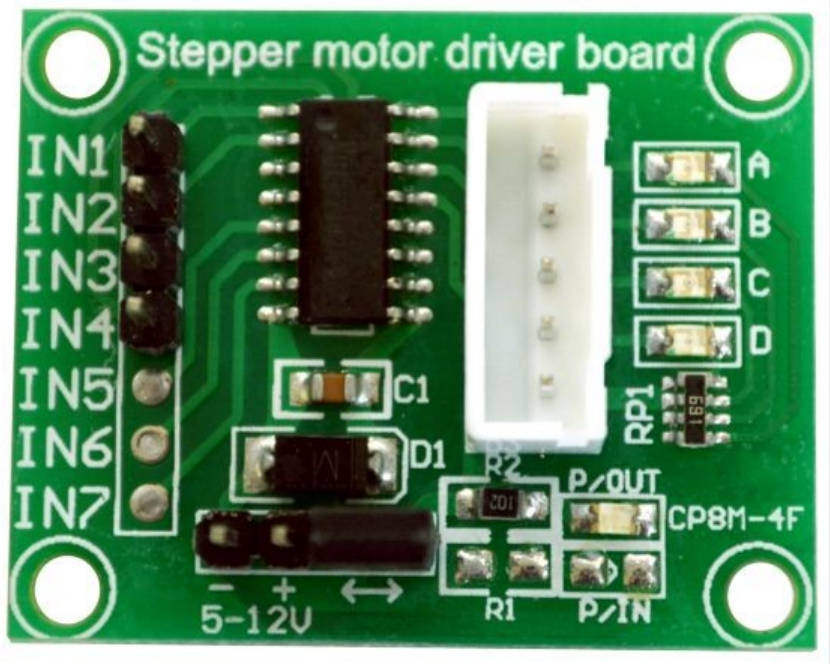
Víctor . Álvarez 201911

Motores PaP – stepper 28BYJ-48

moto.pap.con.libreria.stepper

```
/*
 * Proba o funcionamento dun motor paso a paso
 * usando a librería <Stepper.h> para que dea
 * cinco voltas completas en sentido horario e
 * outras cinco en sentido antihorario.
 */
#include <Stepper.h>

//Pins para control do motor PaP
#define IN1 8
#define IN2 9
#define IN3 10
#define IN4 11

//Parámetros do motor, pasos totais e por fase
#define STEPS 4076 //Pasos totais por volta
#define NSTEPS 8 //Número de pasos por fase (4 en 1 e 2-fase)

int veloc = 45;        //Velocidade do motor en rpm

Stepper motor(STEPS, IN1, IN2, IN3, IN4);

void setup() {
  motor.setSpeed(veloc);
  Serial.begin(9600);
}

void loop() {
  //Ponhemos a andar 5 voltas en sentido horario
  for(int i = 0; i < 5; i++) {
    Serial.print(String(i) + "a volta en sentido horario");
    motor.step(STEPS);
    delay(1000);
  }
  //Ponhemos a andar 5 voltas en sentido antihorario
  for(int i = 0; i < 5; i++) {
    Serial.print(String(i) + "a volta en sentido antihorario");
    motor.step(-STEPS);
    delay(1000);
  }
}
```

- O script do lado fai exactamente o mesmo que o anterior, pero usando a librería <Stepper.h>
- O procedemento motor.step() é bloqueante, é dicir o script non continua mentres o motor non remate o xiro indicado.



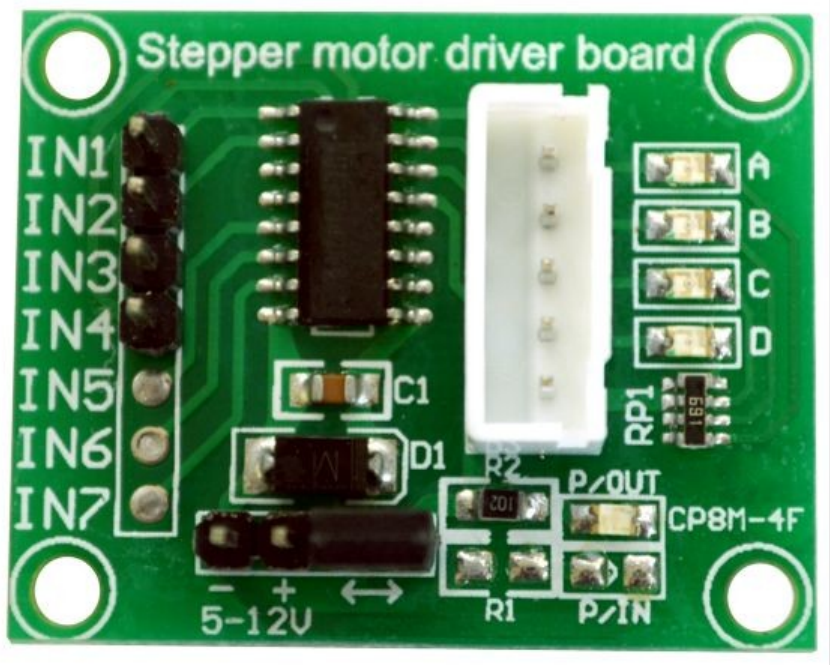
Víctor . Álvarez 201911

Motores PaP – stepper 28BYJ-48

moto.pap.con.libreria.stepper.e.potenciometro

```
/*
 * Proba o funcionamento dun motor paso a paso
 * usando a librería <Stepper.h> para que dea
 * voltas en sentido horario ou antihorario en
 * función do ángulo dun potenciómetro.
 *
 * O ángulo do potenciómetro tamén regula
 * o valor absoluto da velocidade.
 */
#include <Stepper.h>

//Pins para control do motor PaP
#define IN1 8
#define IN2 9
#define IN3 10
#define IN4 11

//Pin para a lectura do potenciómetro
#define POT A0

//Parámetros do motor, pasos totais e por fase
#define STEPS 4076 //Pasos totais por volta
#define NSTEPS 8 //Número de pasos por fase (4 en 1 e 2-fase)

int veloc = 45;         //Velocidade do motor en rpm
int vPotenciometro = 0; //Lectura potenciómetro

Stepper motor(STEPS, IN1, IN2, IN3, IN4);

void setup() {
  motor.setSpeed(veloc);
  Serial.begin(9600);
}

void loop() {
  //Lemos o potenciómetro e mapeamos a [-STEPS, STEPS]
  vPotenciometro = map(analogRead(POT), 0, 1023, -STEPS, STEPS);
  motor.step(vPotenciometro);
  delay(800);
}
```

- Este novo script move un número de pasos, determinado por un potenciómetro, en sentido horario ou en sentido antihorario.
- O procedemento motor.step() é bloqueante, é dicir o script non continua mentres o motor non remate o xiro indicado.



Víctor . Álvarez 201911

Motores PaP

- Nesta unidade aprendemos a:
  - usar un motor paso a paso 28BYJ-48 e conexionar empregando un driver UNL2003,
  - programar o avance e retroceso do motor manualmente, empregando a secuencia correcta de de activación/desactivación dos pins,
  - usar a librería <Stepper.h> para controlar o stepper,
  - combinar outro sensor (potenciómetro), para modificar avance/retroceso, número de pasos e velocidade do motor.

Víctor . Álvarez 201911