Féidearthachtaí as Cuimse
Infinite Possibilities

Week 9
Arduino programming

Fundamentals of IoT
Dr. Eoin Rogers (eoin.rogers@tudublin.ie)

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TU DUBLIN
TECHNOLOGICAL UNIVERSITY DUBLIN

Lesson Outline

- Introducing the Arduino and how to program it
- We won’t be using the Arduino directly this year, but there is one in the kits that you can play around with if you’d like (I have put up lab exercises in case you want to try them!)
- Knowing how to program it might be useful for IoT FYPs and other projects (although be aware of the limitations of the Arduino!)

Arduino

A widely used, open-source SBC

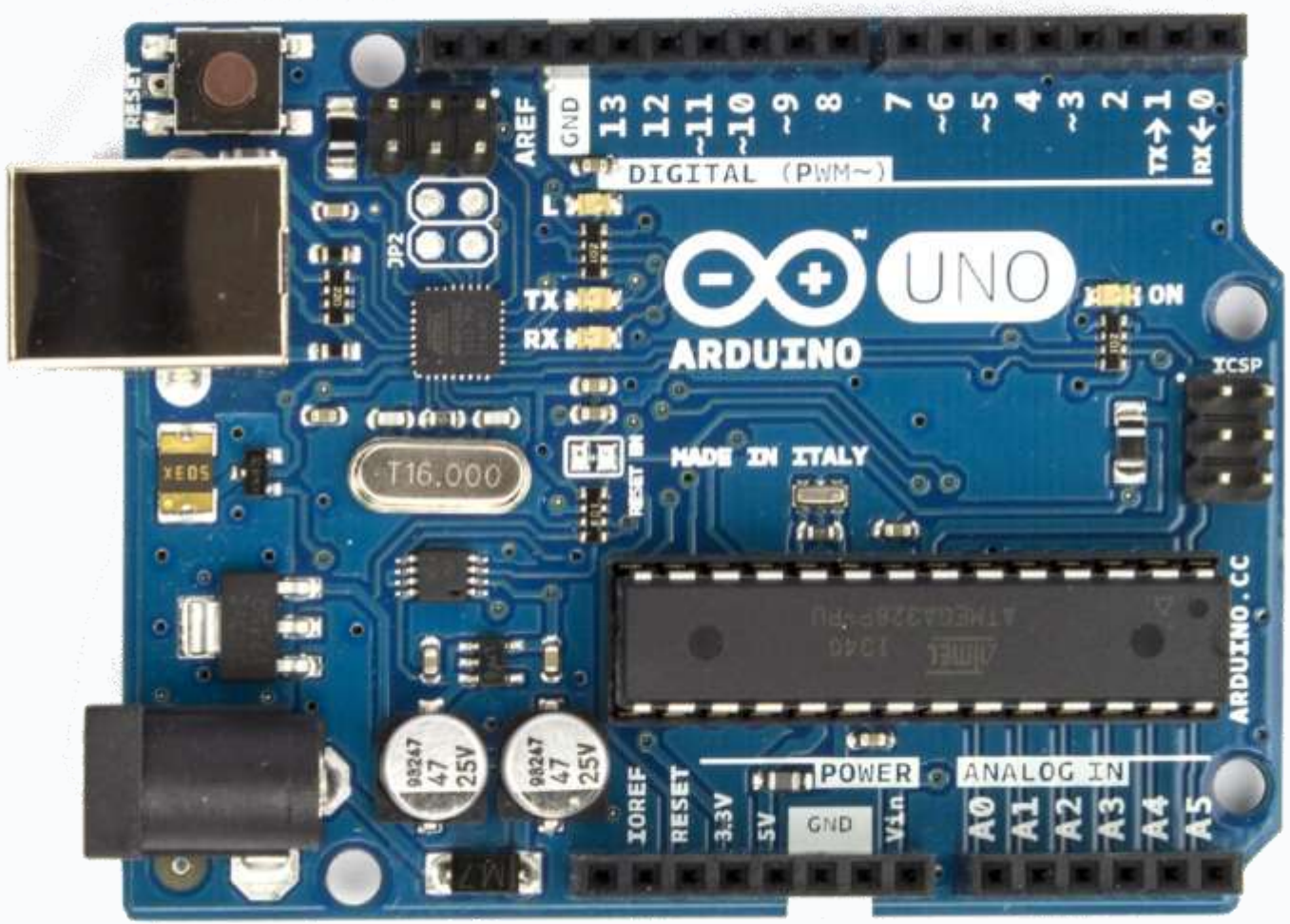
What is an Arduino

- An open-source **single-board computer** (**SBC**)
- Powered by an **8-bit microcontroller** (Atmega 328P)
- Developed by a group based in Italy
- Each cost about €20
- Programming in a dialect of C++

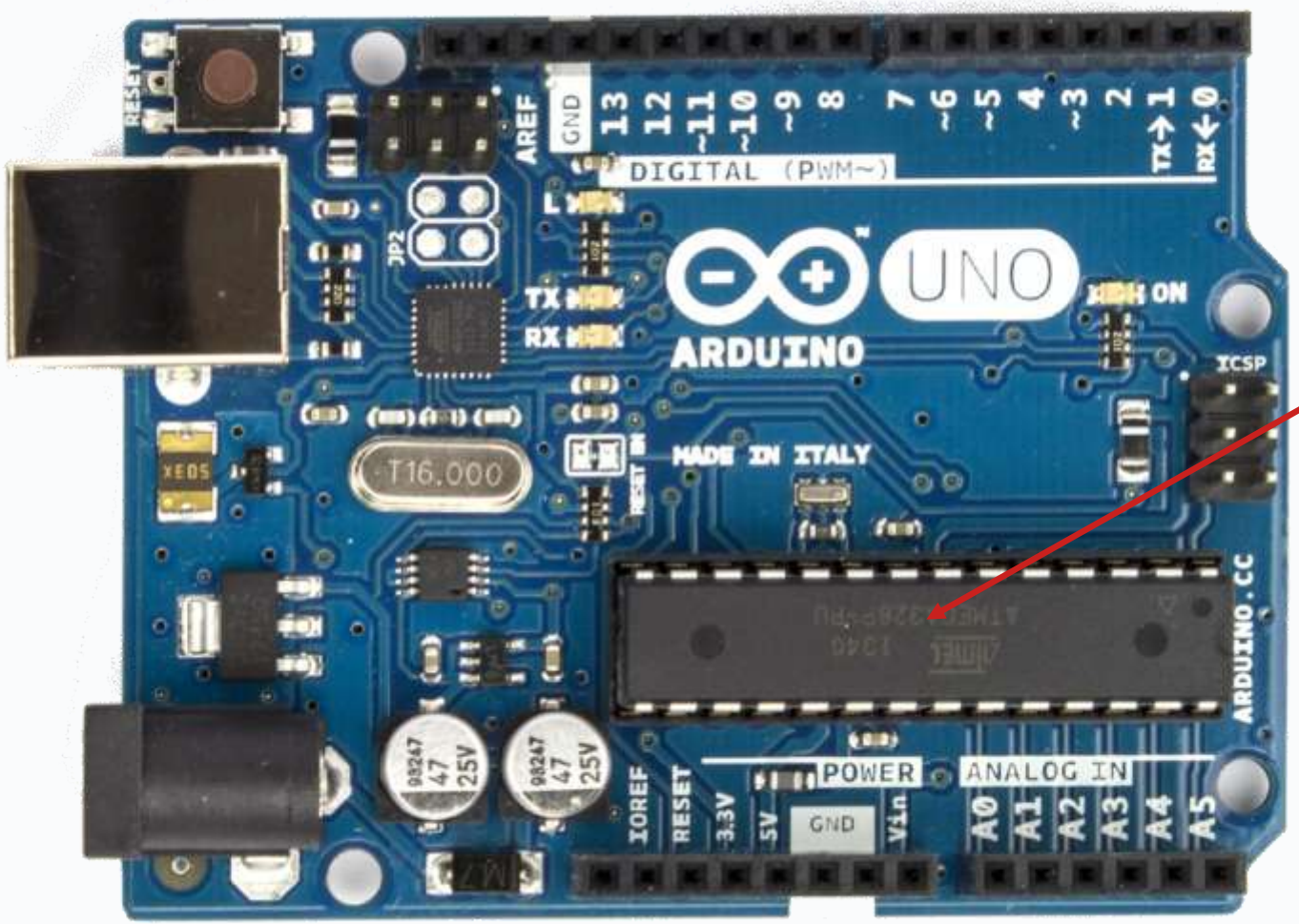
Advantages of the Arduino

- Very cheap
- Huge range of models
- Hugely extensible
  - A family of add-on-boards to expand Arduinos – called shields – are widely used

Arduino Uno

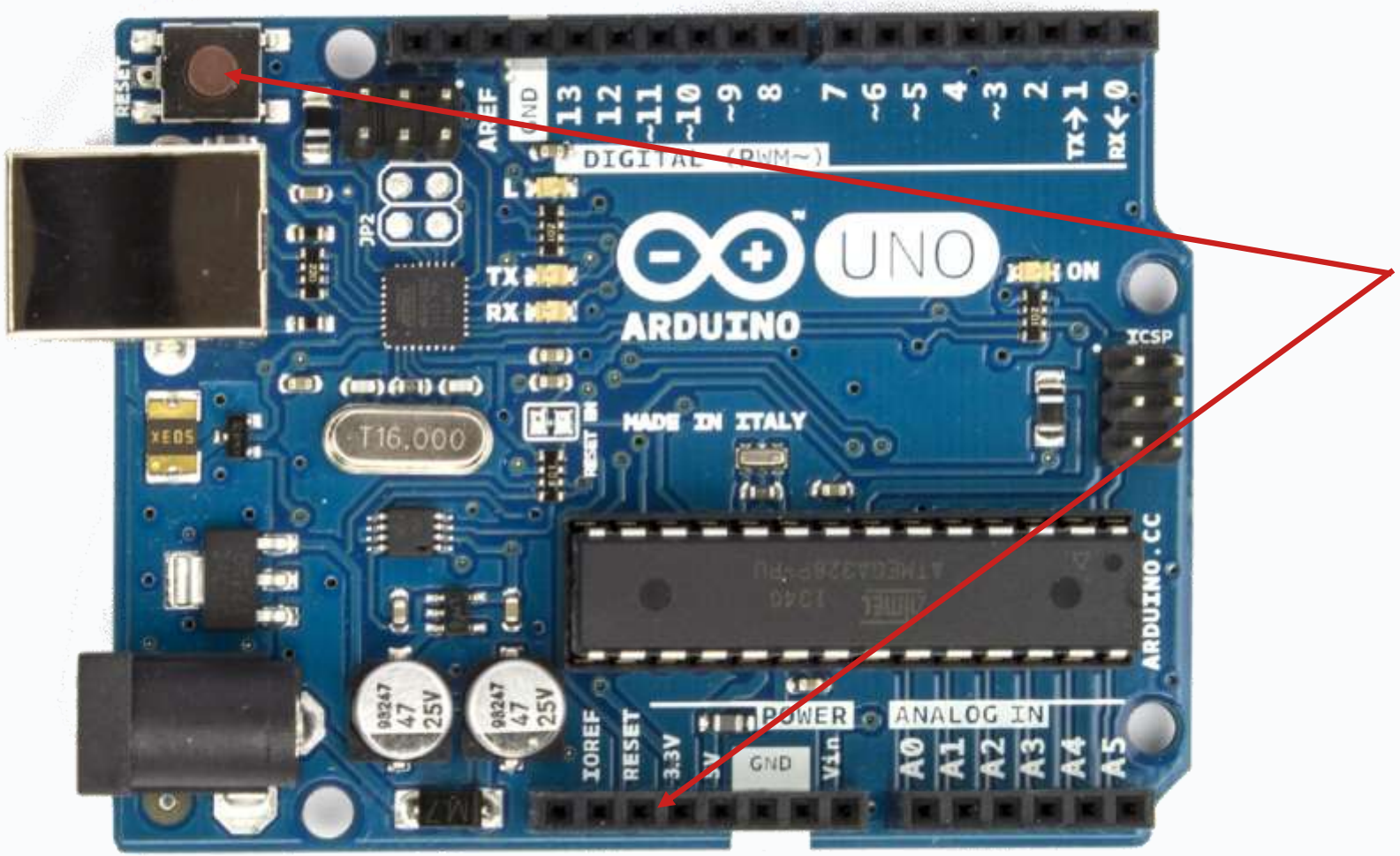

DIGITAL (PWM~)
UNO
ARDUINO
MADE IN ITALY
POWER
ANALOG IN
ARDUINO.CC
ICSP

Arduino Uno



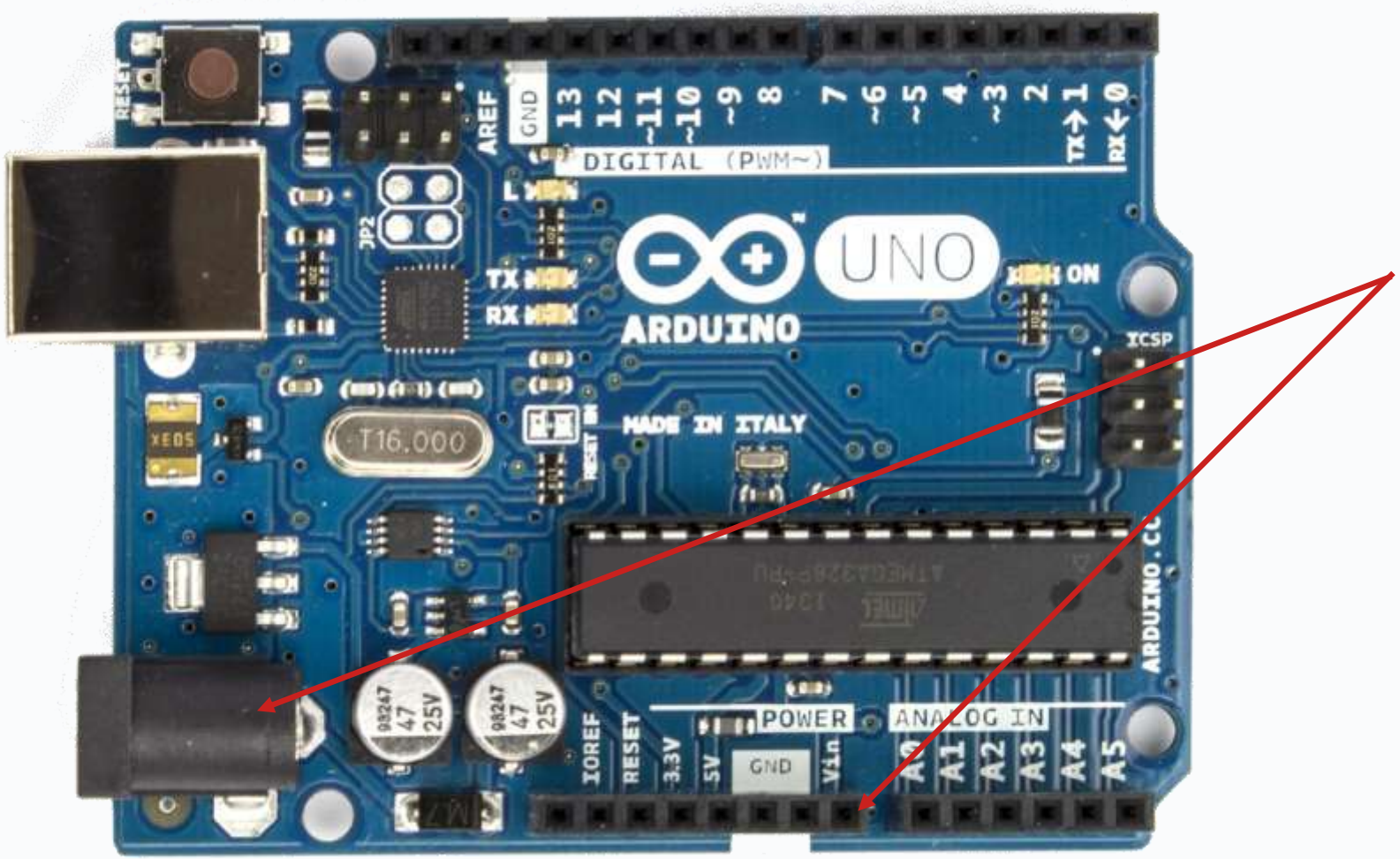
Microcontroller

Arduino Uno



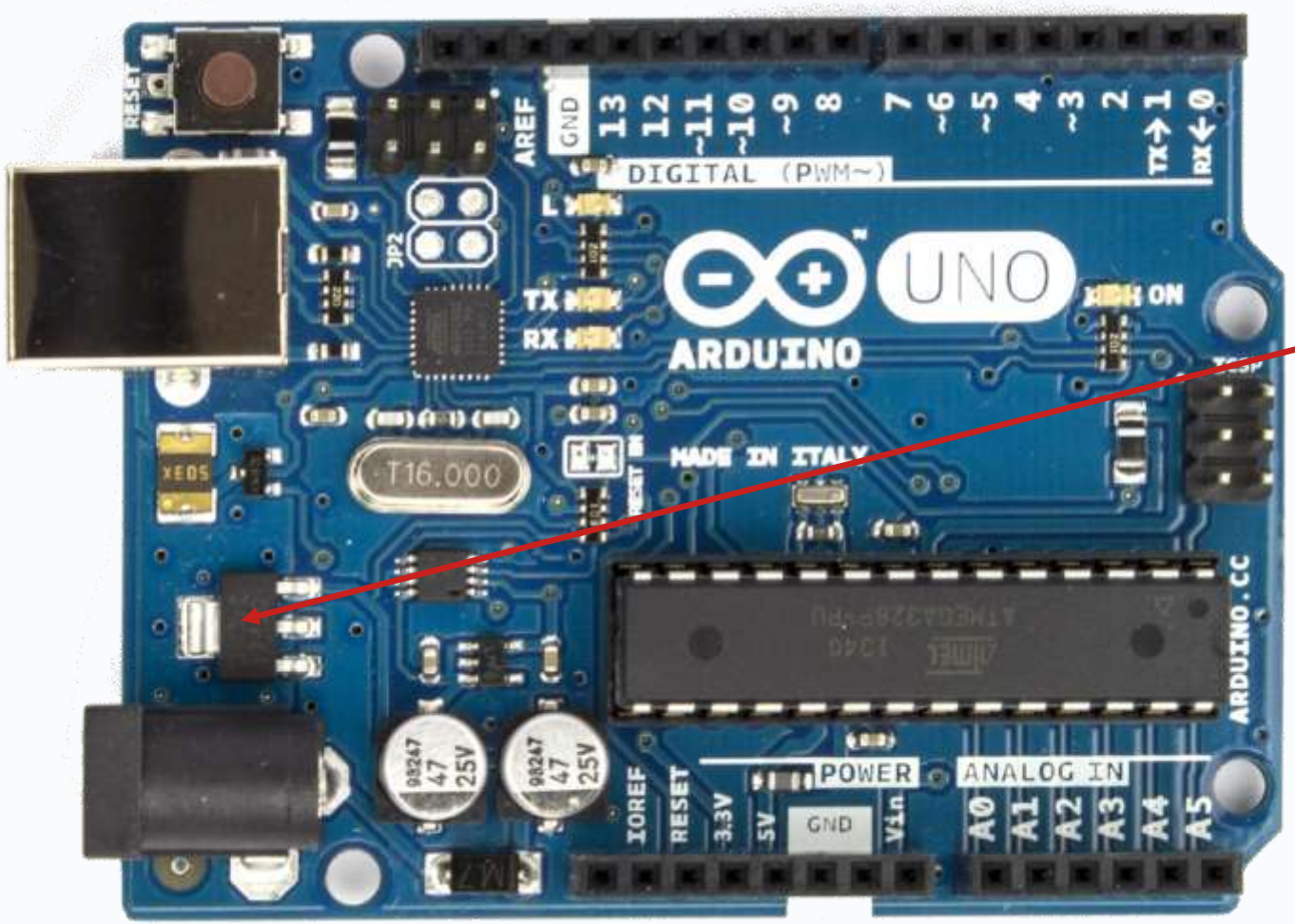
Reset button

Arduino Uno



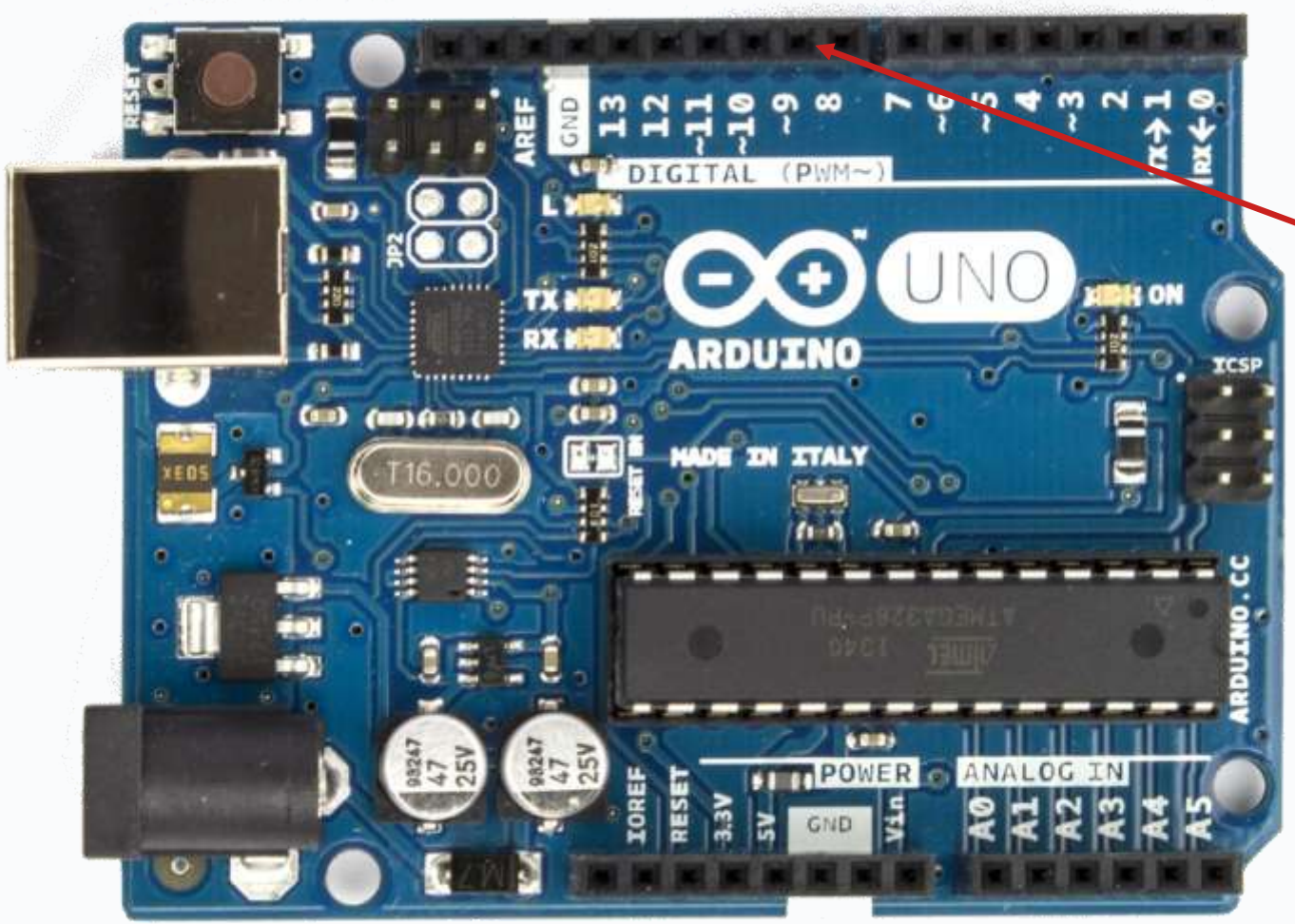
Power

Arduino Uno



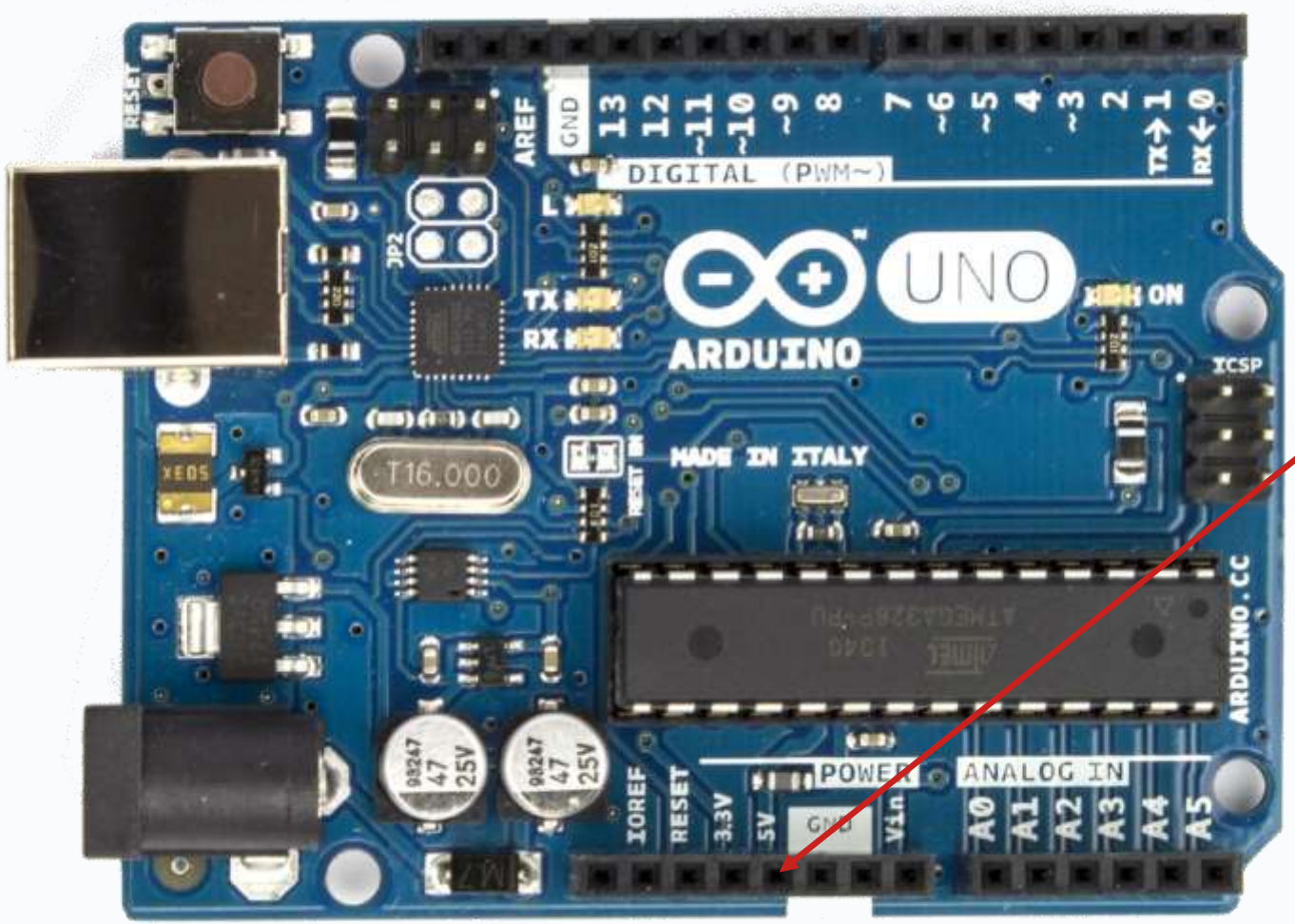
Voltage regulator

Arduino Uno



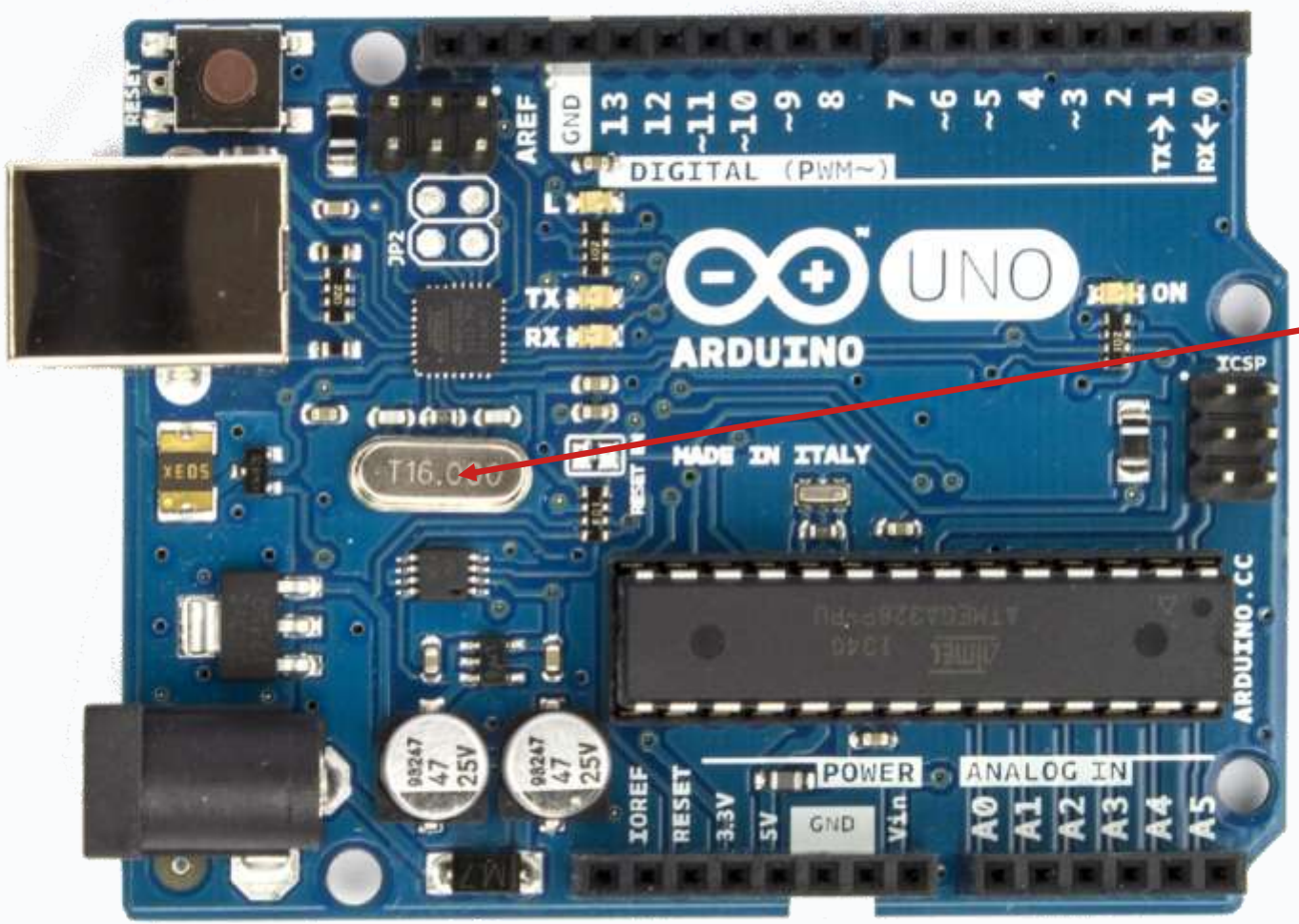
Digital I/O

Arduino Uno



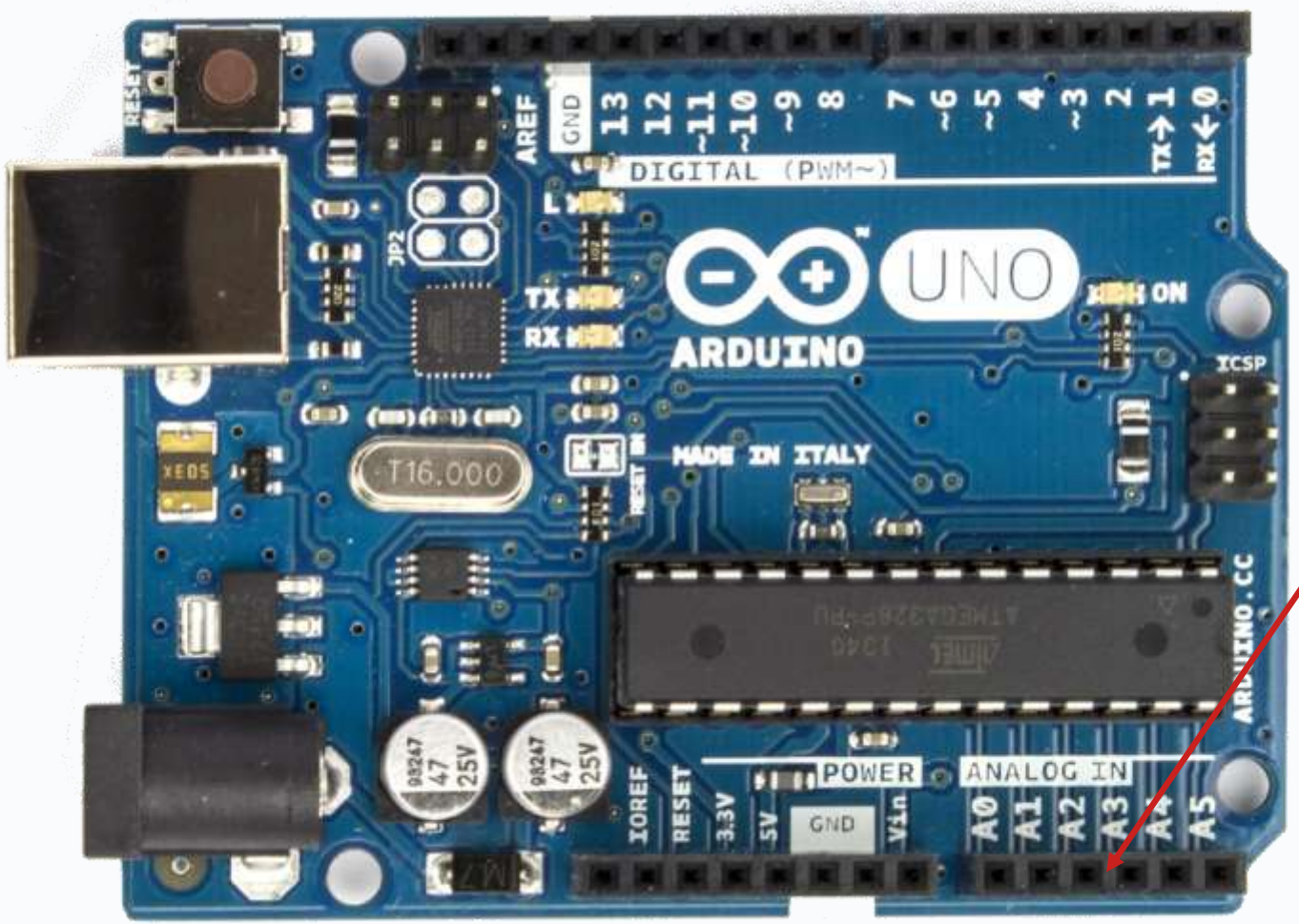
Peripheral power

Arduino Uno



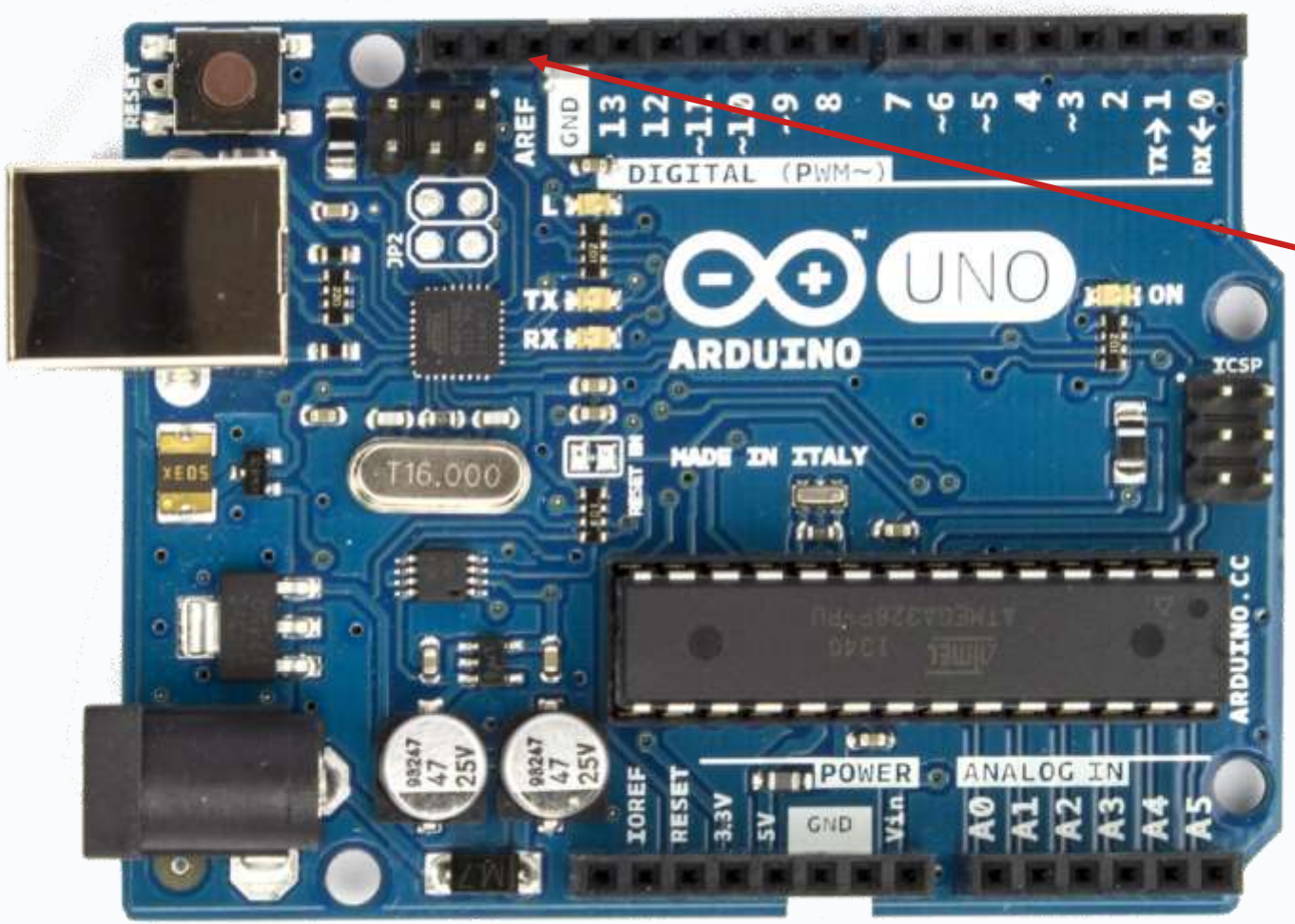
Oscillator

Arduino Uno



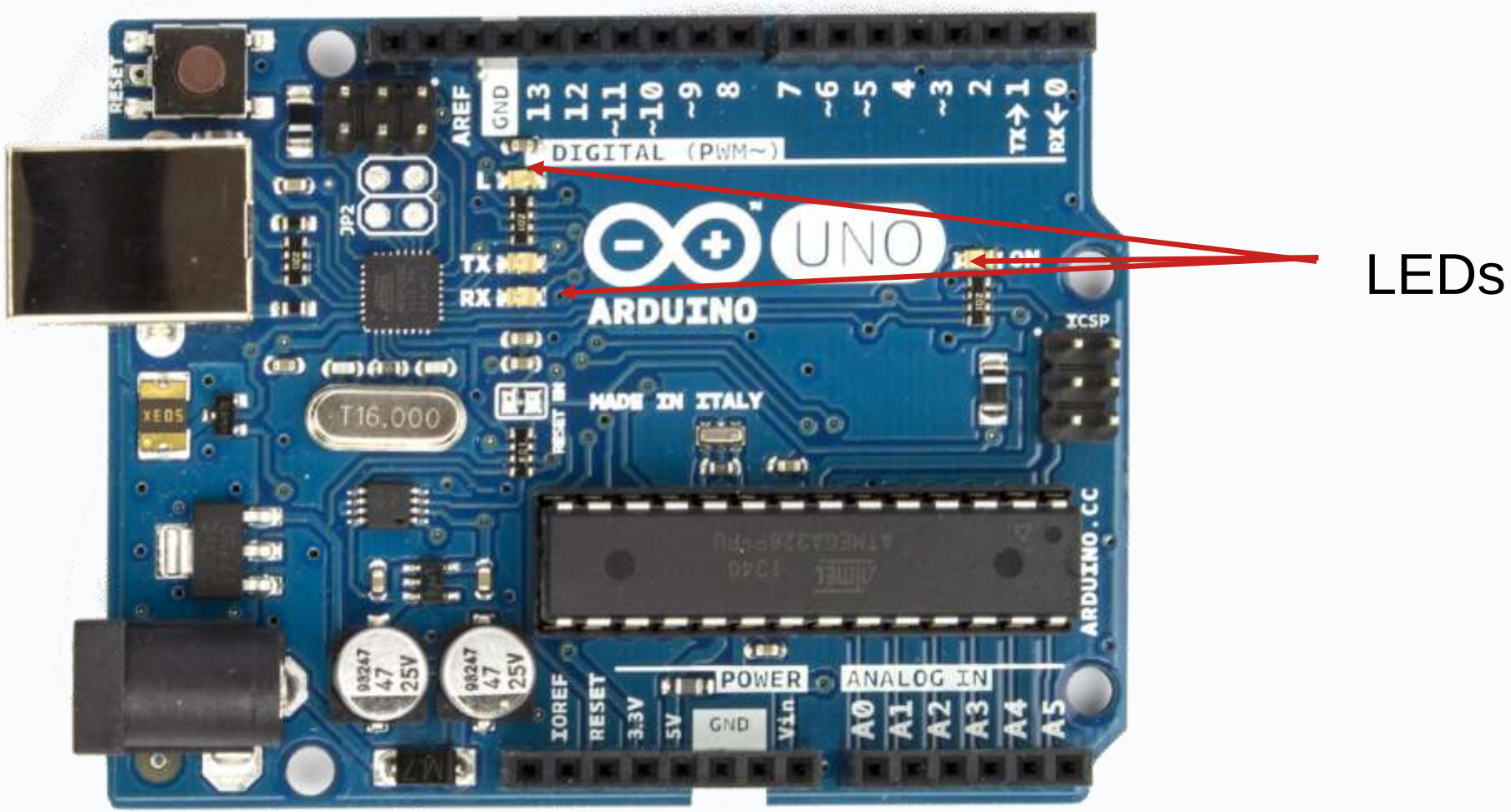
Analogue input

Arduino Uno



AREF pin

Arduino Uno


LEDs
RESET
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ON
UNO
ARDUINO
ICSP
MADE IN ITALY
T16.000
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
Vin
A0
A1
A2
A3
A4
A5
ARDUINO.CC

Programming the Arduino

There is an Arduino IDE which makes this very easy!

Sketches

- Arduino programs are called sketches
- They are typically written in a variant of C++ (basically C++ for AVR with a non-default linker script)
- They are transferred to the Arduino via the USB port

Basic sketch

- An Arduino program should have two functions:
  - **setup()**, which contains initialisation code
  - **loop()**, which runs continuously until the Arduino is turned off!
- Unlike a typical C++ program, there is **no main()**

Blinking an LED

```
#define LED_PIN 13

void setup() {
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    digitalWrite(LED_PIN, HIGH);
    delay(1000);
    digitalWrite(LED_PIN, LOW);
    delay(1000);
}
```

Blinking your own LED

```
#define LED_PIN 9

void setup() {
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    digitalWrite(LED_PIN, HIGH);
    delay(1000);
    digitalWrite(LED_PIN, LOW);
    delay(1000);
}
```

Making the LED fade in and out

```
#define LED_PIN 9

void setup() {
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    for (int brightness = 0; brightness <= 255; brightness += 5) {
        analogWrite(LED_PIN, brightness);
        delay(20);
    }

    for (int brightness = 255; brightness >= 0; brightness -= 5) {
        analogWrite(LED_PIN, brightness);
        delay(20);
    }
}
```

Potentiometer diagram

Potentiometer code

```
#define LED_PIN 9

void setup() {
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    int brightness = analogRead(A0);
    brightness = map(brightness, 0, 1023, 0, 255);
    analogWrite(LED_PIN, brightness);
}
```

Button

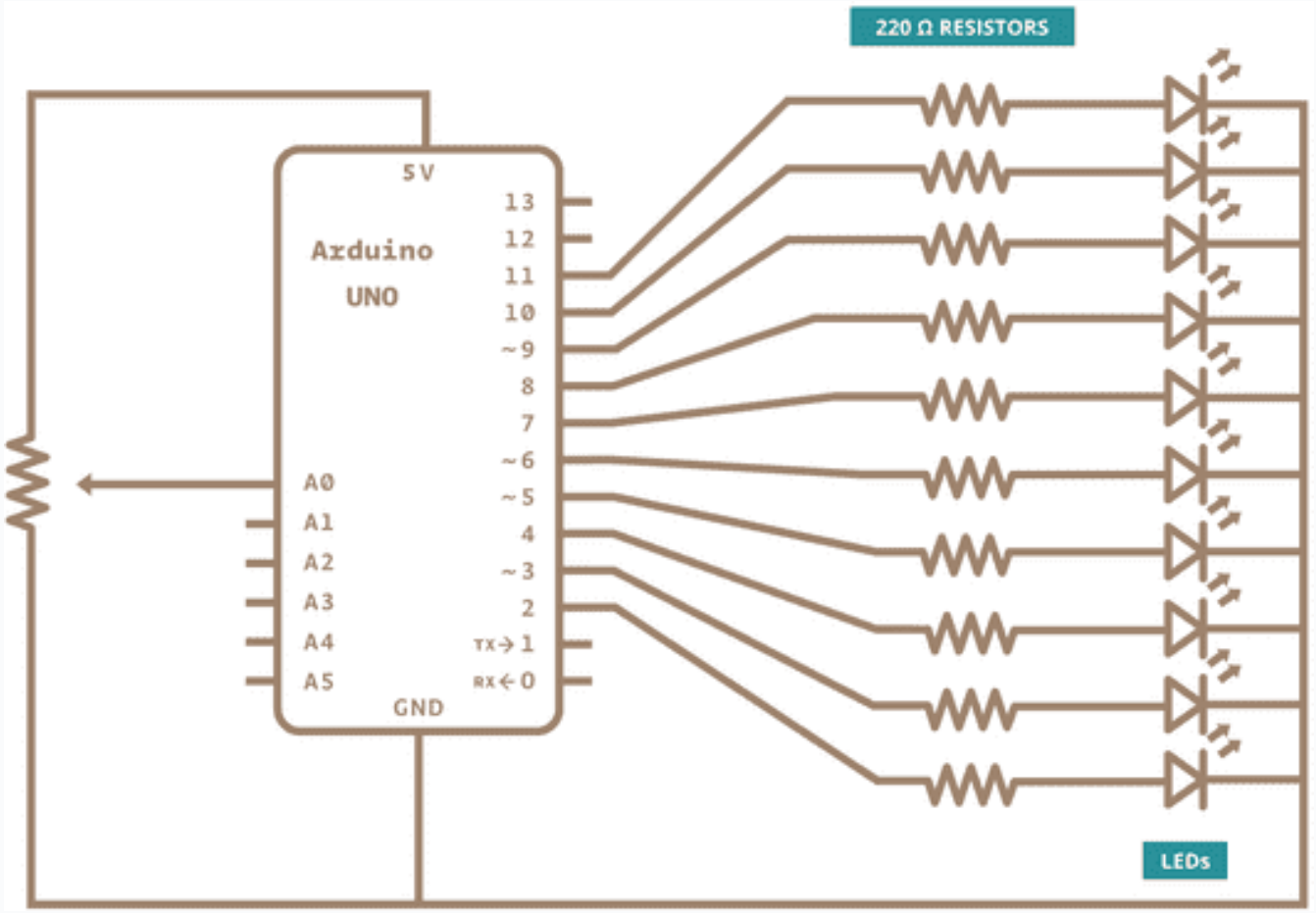
```
#define LED_PIN 9
#define BUTTON_PIN 2

void setup() {
    pinMode(LED_PIN, OUTPUT);
    pinMode(BUTTON_PIN, INPUT);
}

bool led_on = false;
```

```
void loop() {
    int button_state = digitalRead(BUTTON_PIN);

    if (button_state == LOW)
        return;

    if (led_on) {
        digitalWrite(LED_PIN, LOW);
        led_on = false;
    } else {
        digitalWrite(LED_PIN, HIGH);
        led_on = true;
    }
}
```

Serial monitor (useful for debugging)

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

void loop() {
    Serial.println("Hello, World!");
    delay(1000);
}
```

LED bar chart diagram



LED bar chart code

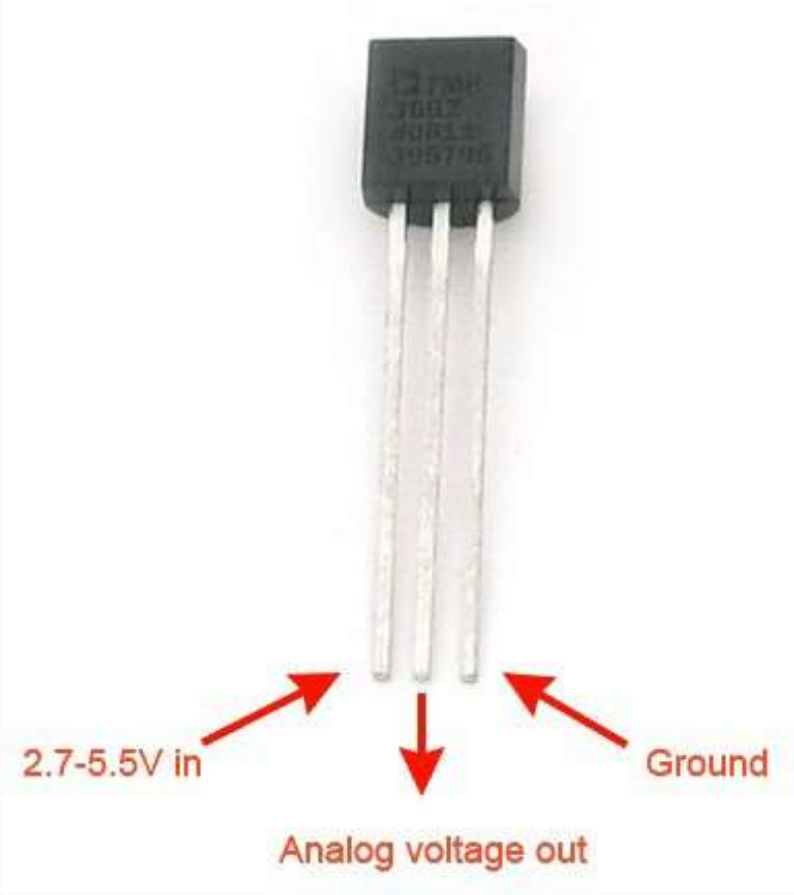
```
#define FIRST_LED 2
#define LAST_LED 10

const int num_leds = LAST_LED - FIRST_LED + 1;

void setup() {
    for (int i = FIRST_LED; i <= LAST_LED; i++)
        pinMode(i, OUTPUT);
}
```

```
void loop() {
    int potentiometer = analogRead(A0);
    potentiometer = map(potentiometer, 0, 1023, 0, num_leds);
    int count = 0;
    for (int i = FIRST_LED; i < LAST_LED; i++)
        if (count < potentiometer) {
            digitalWrite(i, HIGH);
            count++;
        } else {
            digitalWrite(i, LOW);
        }
}
```

Servos

```
#include <Servo.h>

Servo servo;
int angle = 0;

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

void loop() {
    servo.write(angle);
    delay(1000);
    angle = angle ? 0 : 180;
}
```

TMP36 sensor



TMP36 behaviour

- The analogue output pin outputs a voltage between 0 and 5 volts (0 and 5000 millivolts)
- The temperature (in °C) is:
  - $(V_{out} – 500) / 10$

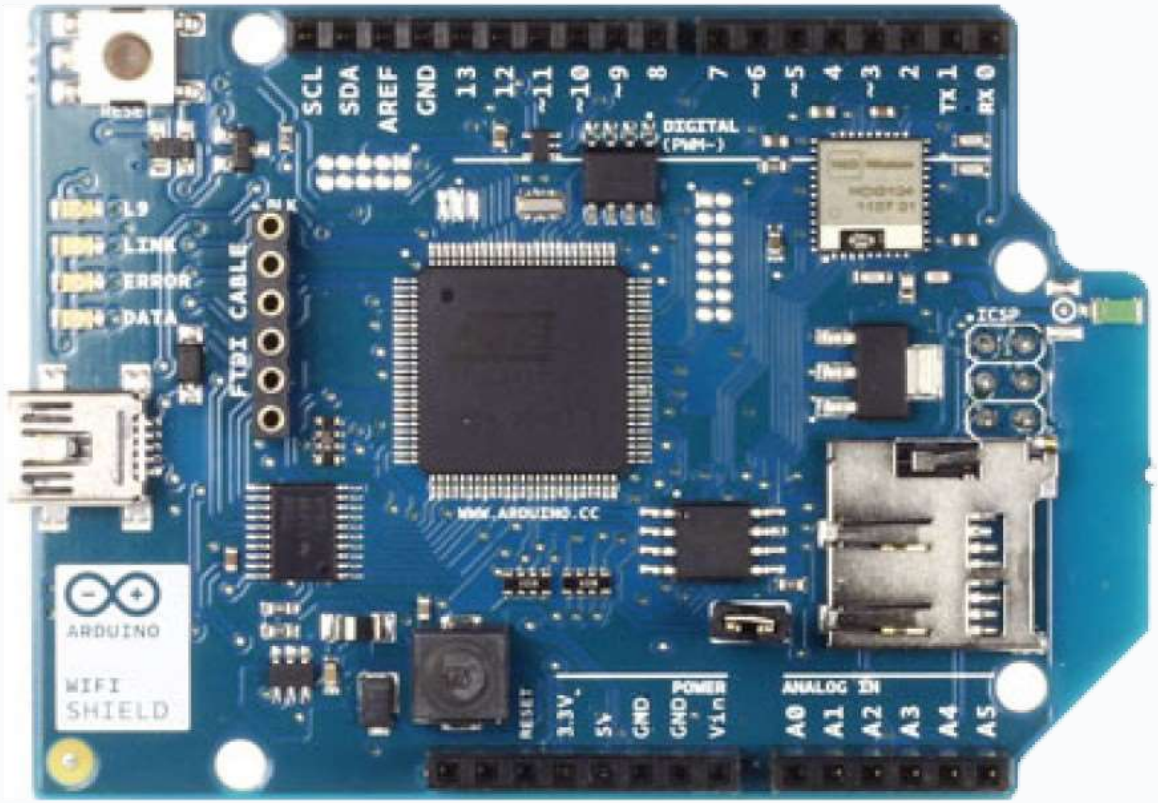
TMP36 code

```
#define TMP36_PIN A0

void setup() {}

void loop() {
    int temperature = analogRead(TMP36_PIN);
    temperature = map(temperature, 0, 1023, 0, 5000);
    float celsius = ((float)temperature - 500) / 10;
}
```

Networking

We can’t really do this in TU Dublin

Networking shield



Connecting to WiFi

```
#include <WiFi.h>

char *ssid = "network name";
int status = WL_IDLE_STATUS;

void setup() {
    Serial.begin(9600);
    status = WiFi.begin(ssid);

    if (status != WL_CONNECTED)
        Serial.println("Can't connect to wifi :-(");
    else
        Serial.println("Connected to wifi :-D");
}
```

```
void loop() {
    if (status == WL_CONNECTED) {
        // Do stuff...
    }
}
```

MQTT subscriber

```
#include <WiFi.h>
#include <MQTT.h>

const char *ssid = "network name";
const char *pass = "network password";
const char *broker = "broker";

MQTTClient mqtt;
WiFiClient wifi;

void msgHandler(
    String &topic,
    String &payload
) {
    // Handle the topic and
    // payload here...
}
```

```
void setup() {
    WiFi.begin(ssid, pass);
    mqtt.begin(broker, wifi);
    mqtt.onMessage(msgHandler);
}

void loop() {
    mqtt.loop();
}
```

Summary

- Introduced the Arduino and its programming model
- Provided a number of examples of Arduino sketches
- Described the TMP36 temperature sensor
- Gave a basic outline of the Arduino networking API

That’s all for this week

Thanks for your attention!