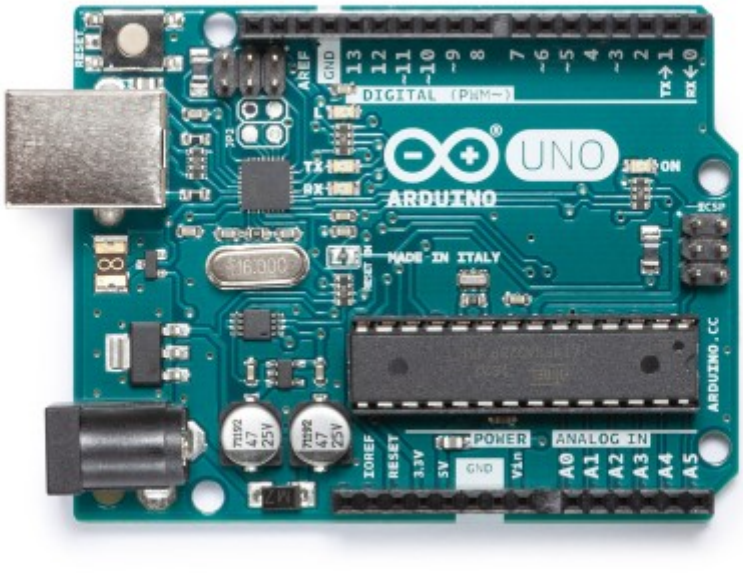
4 Arduino UNO

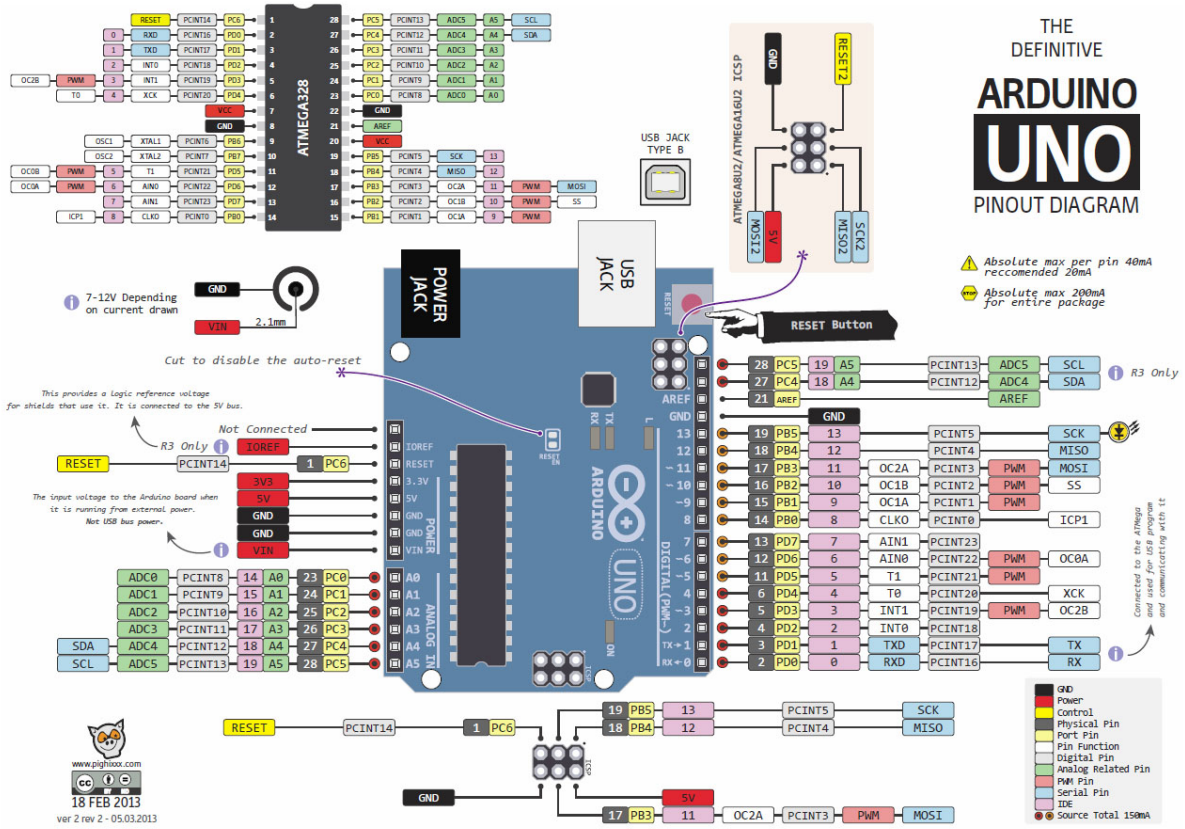
Arduino Uno je:

> The UNO is the best board to get started with electronics and coding. If this is your first experience tinkering with the platform, the UNO is the most robust board you can start playing with. The UNO is the most used and documented board of the whole Arduino family. “Arduino Uno Rev3”

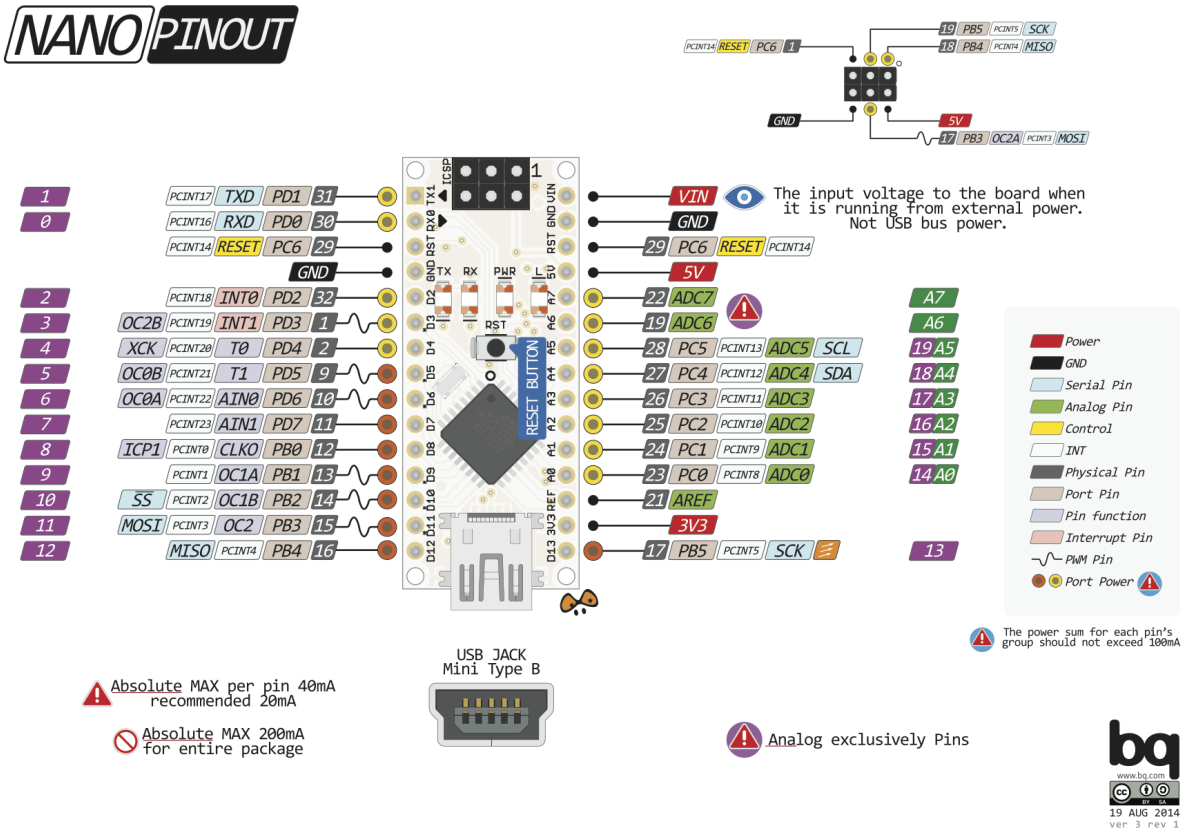


Slika 1: Arduino Uno

4.1 Arduino UNO/NANO pinout



Slika 2: Razporeditev priključkov na krmilniku Arduino UNO (Theman 2015).



Slika 3: Razporeditev priključkov na krmilniku Arduino NANO (Theman 2015).

Kot lahko opazimo, so funkcije priključkov na sliki sl. 2 in sliki sl. 3 enaki. To ni nenavadno, saj sta krmilnika po električni zgradbi enaka, razlikujeta se le v fizični izvedbi.

4.2 Osnove programiranja krmilnika

4.2.1 NALOGA: Nastavitve Arduino IDE programa

Pravilno priključite krminlnik Arduino na računalnik in nastavite programsko okolje Aruino IDE ter sprogramirajte krmilnik s testnim programom **BLINK.INO**. Iz nastavitev programskega okolja prepišite vaše nastavitve za:

1. krmilnik:_____,
2. komunikacijska vrata: _____,
3. mikrokrmilnik: _____.

Testni program BLINK.INO

Nastavite in delovanje krmilnika lahko preverimo s testnim programom **blink.ino**. Na krmilniku Arduino UNO je na priključek **13** priključena svetleča dioda prav za ta namen.

Program 1: Vzorčni program za krmilnik Arduino NANO.

```
1   void setup() {
2     pinMode(13, OUTPUT);
3   }
4
5   void loop() {
6     digitalWrite(13, HIGH);
7     delay(1000);
8     digitalWrite(13, LOW);
9     delay(1000);
10  }
```

4.2.2 NALOGA: Berljivost programske kode

Spremenite prog. 1 tako, da bo čim bolj berljiv. Pri berljivosti programske kode lahko upoštevate neka smernic:

1. Skupek programskih ukazov združujte v funkcije, ki jih smiselno poimenujte.
    1. Iz programske strukture naj se opazi čemu je program namenjen.
    2. Če funkcija `void loop()` bolje opisuje namen programa, jo postavite pred `void setup()`.
    3. Funkcije naj vračajo in sprejemajo smiselne parametre.
2. Uporabljajte `razlagalne spremenljivke in konstante` namesto surovih številčnih vrednosti.
    1. Uporabljene priključke vedno poimenujte npr.: `int LED_PIN = 13;`.
    2. Razmislite o potrebnem območju veljavnosti (angl. scope) spremenljivke.
    3. Vmesne rezultate in vrednosti smiselno poimenujte npr.: `int vrednost_senzorja = analogRead(SENZOR_PIN);`
3. Uporabljajte komentarje LE tam, kjer je to ZARES potrebno.
    1. Imena spremenljivk, konstant in funkcij naj pomagajo pri opisovanju kode, npr.: funkcija `delay(1000)` ne potrebuje nobenega komentarja.
    2. Komentarji lahko postanejo dodatni element, ki ga bo potrebno vzdrževati…

"Arduino Uno Rev3." https://store.arduino.cc/usa/arduino-uno-rev3.

Theman, Dan. 2015. "Can't Get I2C to Work on an Arduino Nano? (Pinout Diagrams)." 2015. https://bigdanzblog.wordpress.com/2015/01/30/cant-get-i2c-to-work-on-an-arduino-nano-pinout-

diagrams.