# Projekat 2

Potrebno je uraditi da na svaki pritisak dugmeta sledeca dioda u nizu zasvijetli. To mozemo uraditi u loop petlji klasicno, tj da stalno provjeravamo da li je taster pritisnut. Da bi oslobodili loop za neke druge stvari koje zelimo da uradimo mozemo korisititi **interupt.**

1. The interrupt vector, which determines what pin can generate an interrupt. This isn't the number of the pin itself - it's actually a reference to where in memory the Arduino processor has to look to see if an interrupt occurs. A given space in that vector corresponds to a specific external pin, and not all pins can generate an interrupt! On the Arduino Uno, **pins 2 and 3** are capable of generating interrupts, and they correspond to interrupt vectors 0 and 1, respectively. For a list of what pins are available as interrupt pins, check out the [Arduino documentation](https://www.arduino.cc/en/Reference/attachInterrupt) on **attachInterrupt()**.

2. The function name of the interrupt service routine - this determines the code that gets run when the interrupt condition is met.

3. The interrupt mode, which determines what pin action triggers an interrupt. The Arduino Uno supports four interrupt modes:

  \* **RISING**, which activates an interrupt on a rising edge of the interrupt pin,

  \* **FALLING**, which activates on a falling edge,

  \* **CHANGE**, which responds to any change in the interrupt pin's value,

  \* **LOW**, which triggers any time the pin is a digital low.

Primjer interrupt-a:

const int buttonPin = 2; // the number of the pushbutton pin

const int ledPin = 13; // the number of the LED pin

// variables will change:

volatile int buttonState = 0; // variable for reading the pushbutton status

void setup() {

// initialize the LED pin as an output:

pinMode(ledPin, OUTPUT);

// initialize the pushbutton pin as an input:

pinMode(buttonPin, INPUT);

// Attach an interrupt to the ISR vector

attachInterrupt(0, pin\_ISR, CHANGE);

}

void loop() {

// Nothing here!

}

void pin\_ISR() {

buttonState = digitalRead(buttonPin);

digitalWrite(ledPin, buttonState);

}

Zna se desiti da zbog raznih smetnji, neidealnih prekidaca Interrupt ocita pritiskanje tastera vise u puta u toku jako kratkog vremena. Zbog toga se koristi tako zvani Debouncing. Ovaj problem se rjesava na dva nacina:

1. Hardverski

2. Softverski

1. Hardverski

Pomocu raznih kondezatora i elektronike…

2. Softverski

Mozemo rijesiti putem funkcije millis, gdje ispitivamo vrijeme koje je proslo od zadnjeg pritiskanja tastera.