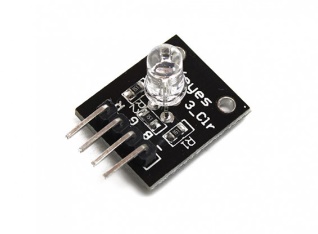
# RGB LED

RGB LED umožňujú zobrazovať rôzne farby. RGB LED má 4 kolíky (red, green, blue, anóda/katóda). Existujú 2 typy RGB LED – so spoločnou anódou alebo katódou.



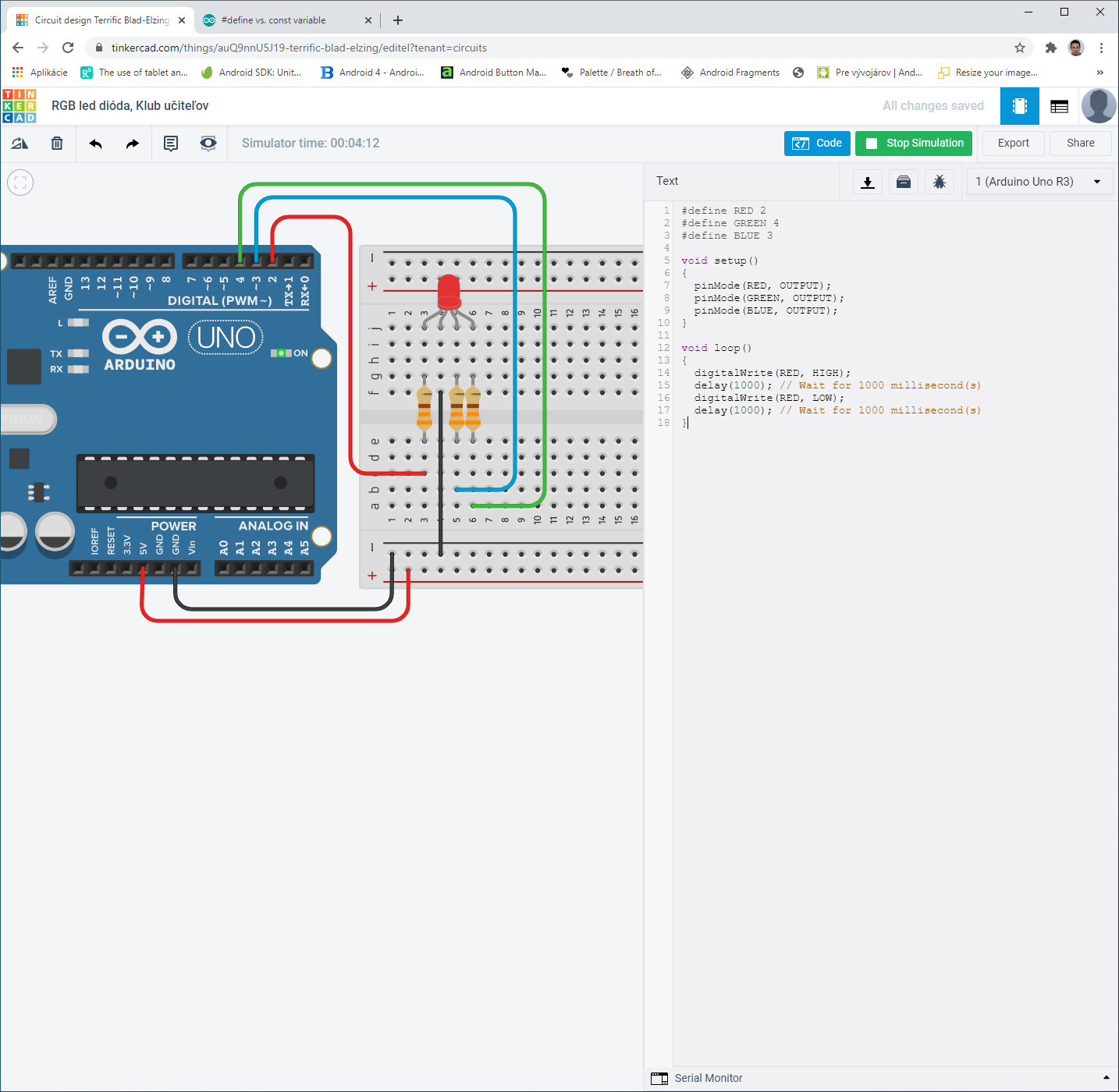
RGB LED súčiastku môžeme kúpiť v dvoch verziách LED so 4mi kolíkmi (obrázok naľavo) a LED na obdĺžnikovej doske (obrázok napravo).

Každá dióda môže mať inú hodnotu odporu, napr. 330ohmov. Hodnotu vypočítame na základe popisu súčiastky. Pomocou PINou pripojených k R, G, B určujeme zapojenie pomocou binárnej sústavy 0 a 1, môžeme ich tiež kombinovať. Každý farba má 256 (0-255) stupňov jasu, pričom môžeme vytvoriť až 2563 rôznych farieb. Každej LEDke môžeme cez kolíku „pustiť“ hodnoty medzi 0 a 255 (piny musia byť napojené na PWM).

Napr.

* R=0, G = 0, B = 0 LED zhasne
* R=1, G = 0, B = 0
* R=1, G = 1, B = 0
* R=0, G = 1, B = 1

**Úloha 1. Rozosvietenie RGB LED.**



#define RED 2

#define GREEN 4

#define BLUE 3

void setup()

{

pinMode(RED, OUTPUT);

pinMode(GREEN, OUTPUT);

pinMode(BLUE, OUTPUT);

}

void loop()

{

digitalWrite(RED, HIGH);

delay(1000); // Wait for 1000 millisecond(s)

digitalWrite(RED, LOW);

delay(1000); // Wait for 1000 millisecond(s)

}

**Úloha 2. Modifikácia príkladu, každú sekundu sa mení farba LEDky.**

(Zapojenie je rovnaké ako v úlohe 1.)

#define RED 2

#define GREEN 4

#define BLUE 3

void setup()

{

pinMode(RED, OUTPUT);

pinMode(GREEN, OUTPUT);

pinMode(BLUE, OUTPUT);

}

void loop()

{

digitalWrite(RED, HIGH);

delay(1000);

digitalWrite(RED, LOW);

delay(1000);

digitalWrite(GREEN, HIGH);

delay(1000);

digitalWrite(GREEN, LOW);

delay(1000);

digitalWrite(BLUE, HIGH);

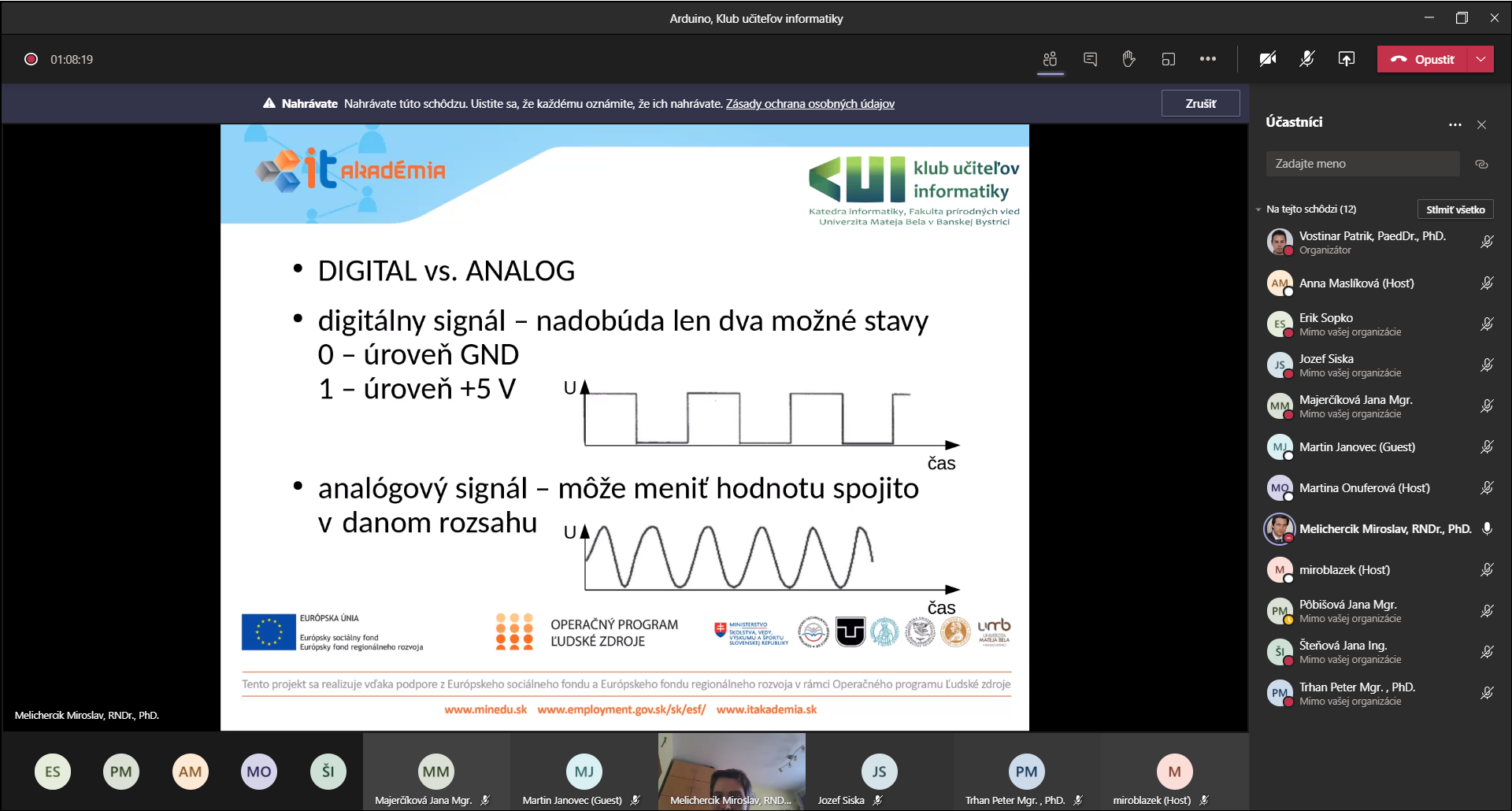
delay(1000);

digitalWrite(BLUE, LOW);

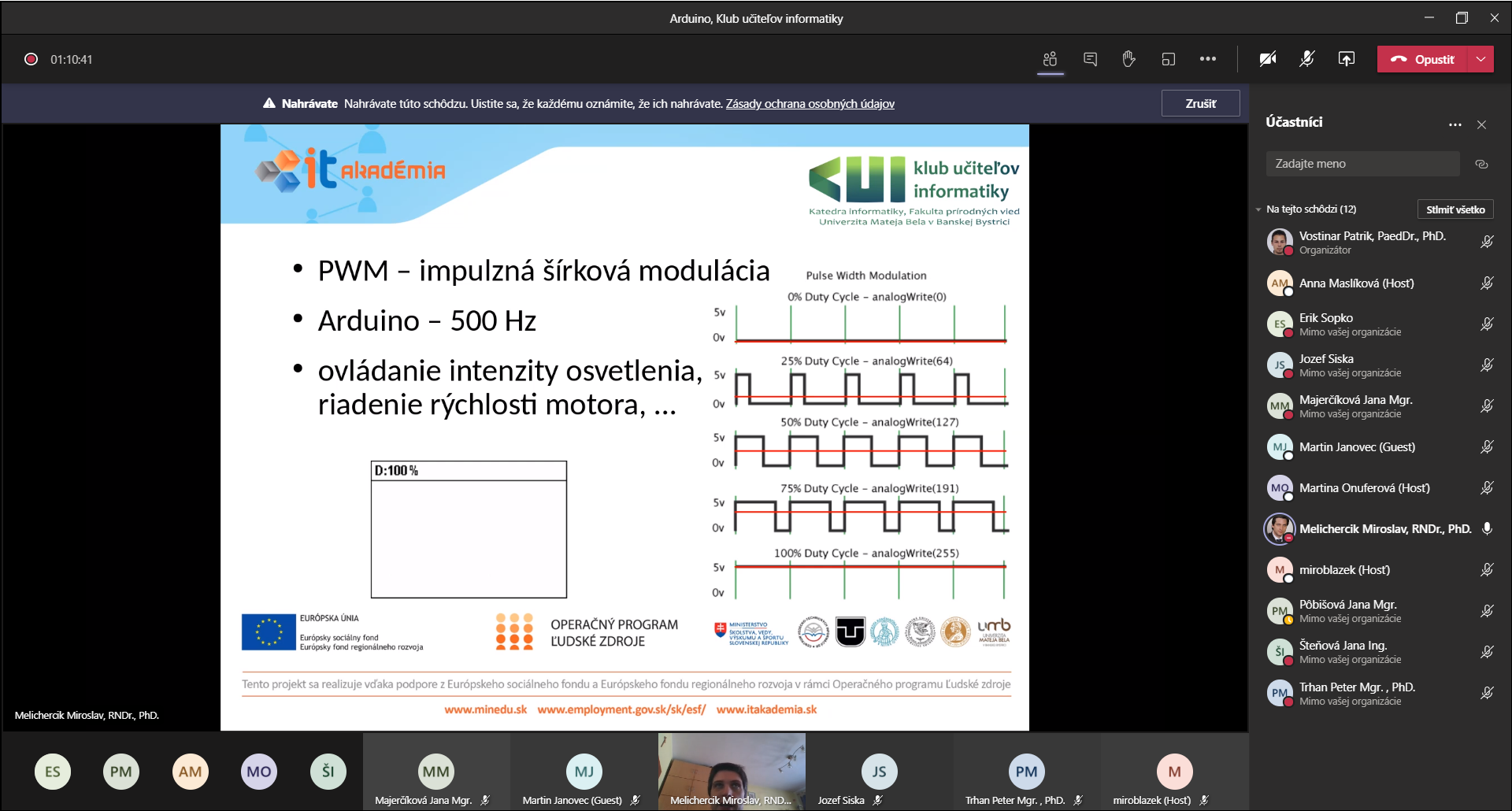
delay(1000);

}

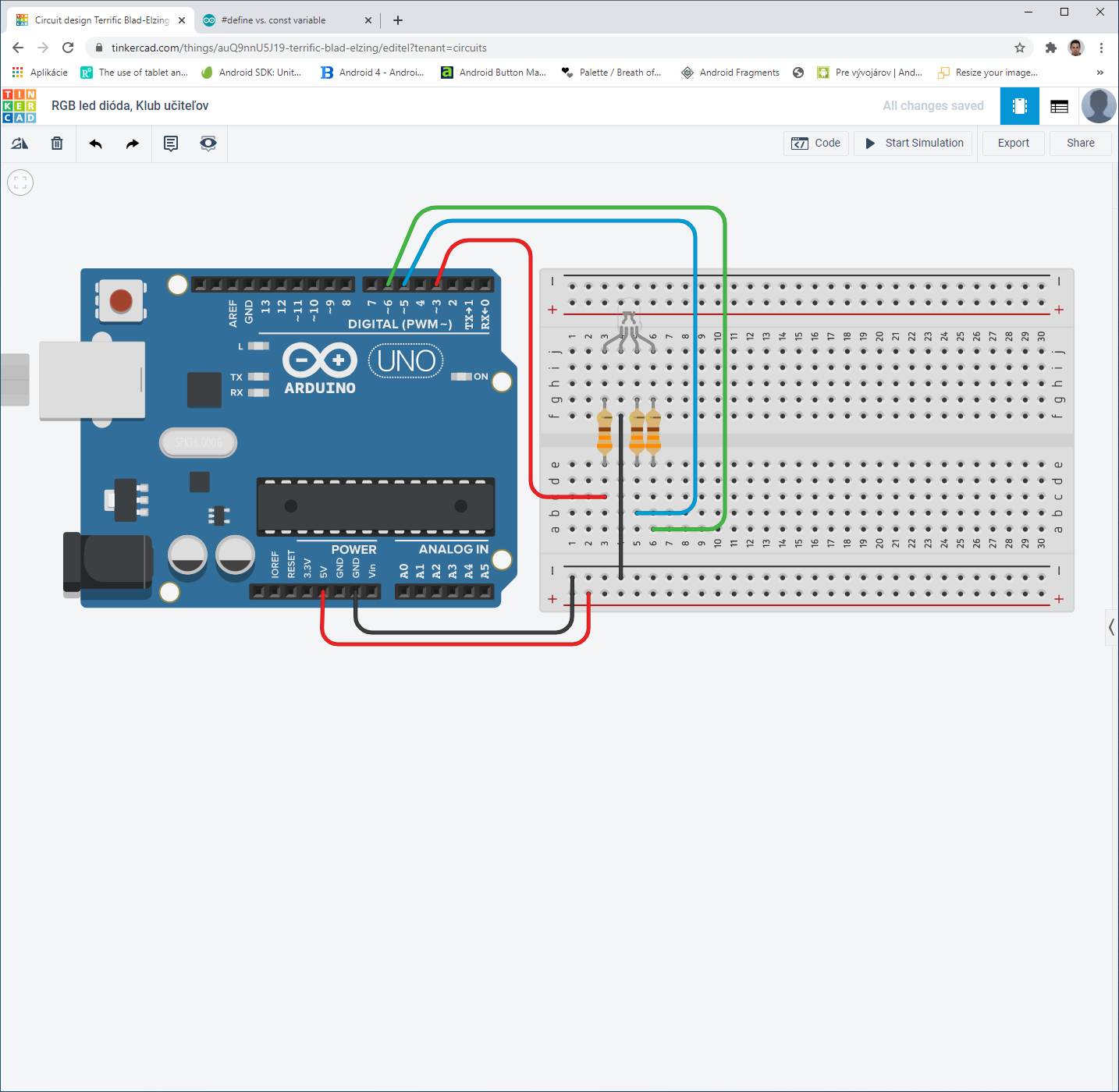
**Digitálny vs Analógový signál**



PWM pracuje na frekvencii 50Hz, jedno zapnutie a vypnutie trvá 2ms. Používame analogWrite(). 8bitové = od 0 do 255.



**Úloha 3. Upravíme pripojenie PINou k arduino – musíme nastaviť PWM PINy na arduinem napr. 3,5,6.**



#define RED 3

#define GREEN 6

#define BLUE 5

void setup()

{

pinMode(RED, OUTPUT);

pinMode(GREEN, OUTPUT);

pinMode(BLUE, OUTPUT);

}

void loop()

{

//kvôli kombináciam dávame cykly do seba

int r,g,b;

for(r=0; r < 256; r+=25) {

analogWrite(RED, r);

for(g=0; g < 256; g+=25) {

analogWrite(GREEN, g);

for(b=0; b < 256; b++) {

analogWrite(BLUE, b+=25);

delay(10);

}

}

}

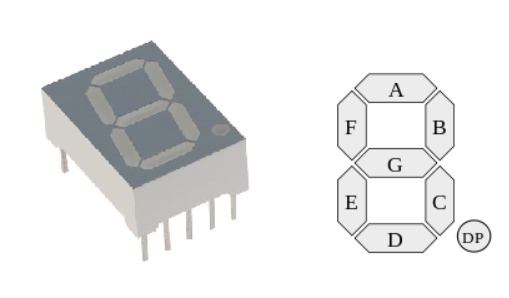
}

Modifikácia príkladu– analogWrite(RED, r) a analogWrite(GREEN, g) – namiesto parametrov r a g dáme konštantu LOW, v tom prípade bude ukazovať iba modrú farbu – zapínanie, vypínanie.

# 7 segmentový displej

Niekoľko LED svetielok zaliatych v jednom púzdre. Väčšinou je jedna z elektród (anóda/katóda – podľa dokumentácie) spojená s LEDkami.

„Osmička“, ktorá sa používa v pokladnách, rôznych čítačkach, atď. Väčšinou má sedem vývodov pre segmenty čísla, vývod pre bodku a spoločnú anódu/katódu. Zhasínaním a zapínaním jednotlivých segmentov môžeme docieliť zobraziť arabských čísiel , hexadecimálnych čísiel, písmen, znakov.



Na zobrazenie čísla 0, 1, 2 na 7segmentovom displeji musíme rozsvietiť:

* Číslo 0: A,B,C,D,E,F
* Číslo 1: B,C
* Číslo 2: A, B, G, E, D

Napr. cez pole

byte dig[10];

dig[0] = B11111100;

dig[1] = B01100000;

dig[2] = B11011010;

dig[3] = B11110010;

dig[4] = B01100110;

dig[5] = B10110110;

dig[6] = B10111110;

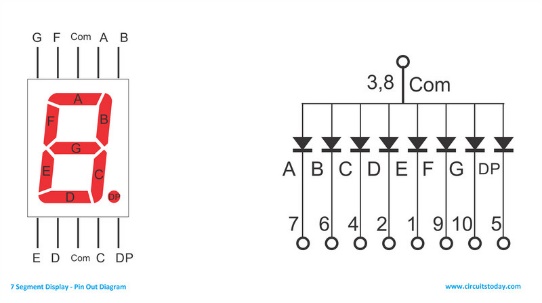
dig[7] = B11100000;

dig[8] = B11111110;

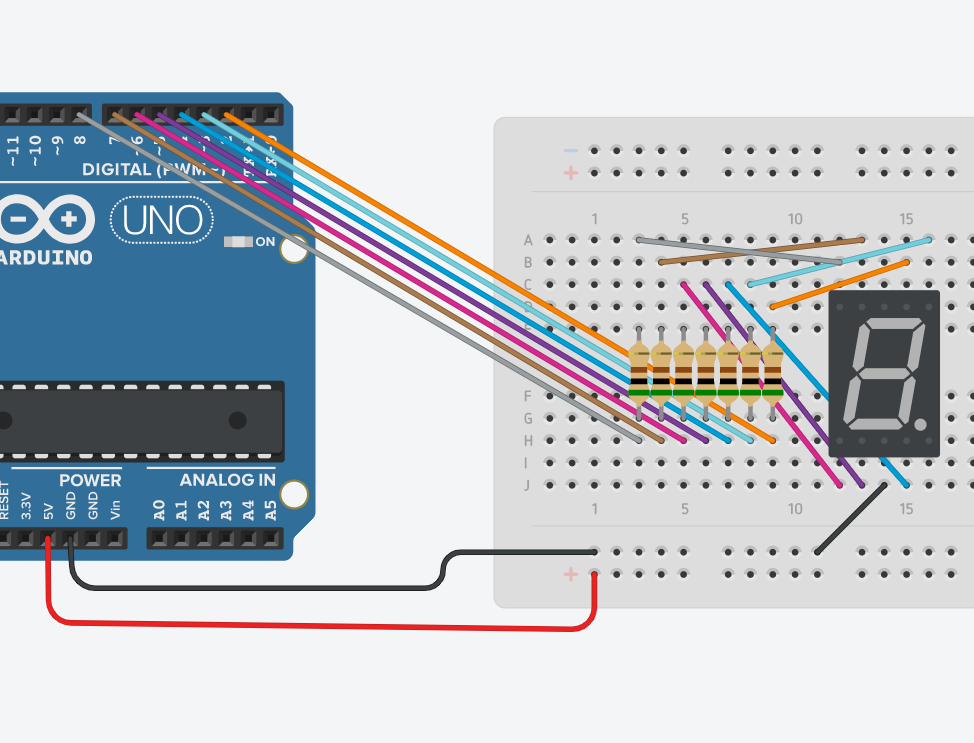
dig[9] = B11110110;

Do breadboardu pridám 7x 500 ohmový odpor, 7segmentový displej (po kliknutí na kolík s názvom Common vyberieme možnosť katóda a súčasne prepojíme PIN Common na GND, zo 7segmentového displeja postupne pripájame piny k digitálnym pinom. Začnem od pinu A a potiahneme káblik k prvému odporu, potupne pokračujem B, C, D, E, F, G. Z druhej strany odporov postupne napojíme odpory na arduino PIN:

* A=2
* B=3
* C=4
* D=5
* E=6
* F=7
* G=8



**Úloha 4. Zapojenie 7segmentového displeja – zobrazenie čísla 8.**



(ak máme spoločnú anódu, tak presne naopak dávame 1 a 0)

void setup()

{

//nastav D2-D8 na OUTPUT

int i;

for (i=2; i<= 8; i++) {

pinMode(i, OUTPUT);

digitalWrite(i, LOW); //zhasne segment

}

}

void loop()

{

byte dig; //abcd efg.

dig = B11111111; //OxFF B 1111 1111

int i;

byte m = B10000000;

for (i=0; i<= 8; i++) {

//rosvietenie alebo zhasnutie Acka

if(dig & m) {

digitalWrite(i+2, HIGH);

} else {

digitalWrite(i+2, LOW);

}

m = m >> 1; //posun bity o 1 vpravo

}

}

/\*

abcdefg.

A

11111111 dig

10000000 maska

10000000 bit sucin

B

11111111 dig

01000000 maska

01000000 bit sucin

//postupne prejdem v3etkz segmenty

\*/

**Úloha 5. Postupné vypisovanie čísiel 0-9**

void setup()

{

//nastav D2-D8 na OUTPUT

int i;

for (i=2; i<= 8; i++) {

pinMode(i, OUTPUT);

digitalWrite(i, LOW); //zhasne segment

}

}

void loop()

{

//pomocou 1 a 0 zhasínam a zapínam segmenty

byte dig[10];

dig[0] = B11111100;

dig[1] = B01100000;

dig[2] = B11011010;

dig[3] = B11110010;

dig[4] = B01100110;

dig[5] = B10110110;

dig[6] = B10111110;

dig[7] = B11100000;

dig[8] = B11111110;

dig[9] = B11110110;

for (int d=0; d< 10; d++) {

byte i, m = B10000000;

for (i=0; i<= 8; i++) {

//rosvietenie alebo zhasnutie Acka

if(dig[d] & m) {

digitalWrite(i+2, HIGH);

} else {

digitalWrite(i+2, LOW);

}

m = m >> 1; //posun bity o 1 vpravo

}

delay(1000);

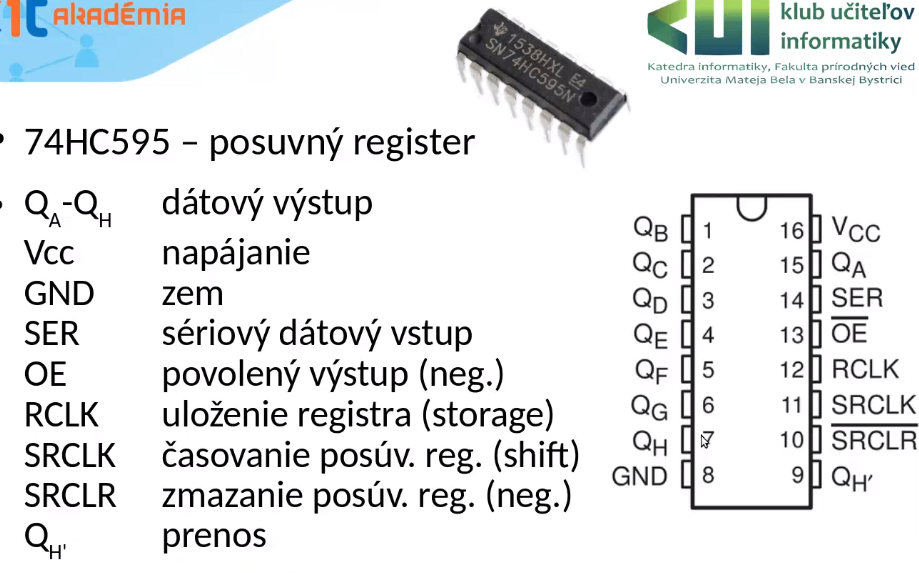
}

}

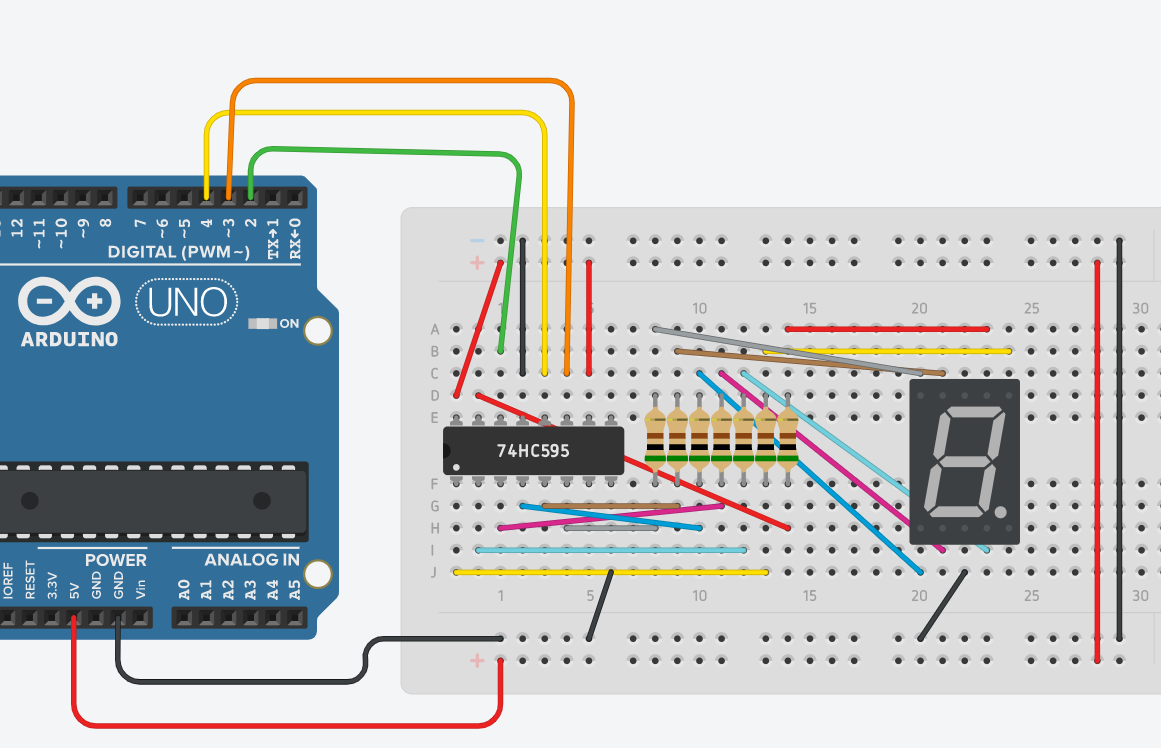
# Integrovaný obvod

V úlohe 4 a 5 sme si ukazovali riadenie 7segmentového displeja pomocou mikrokontroléra Arduino. Ďalšou možnosťou je riadiť niekoľko 7segmentových displejov pomocou použitia posuvných registrov, ktoré pracujú pomocou sériového prenosu dát. V príklade použijeme posuvný register 74HC595. Posuvný register je sústava obvodov, ktorými sa logická informácia posúva pomocou hodinového impulsu. Potrebujeme:

* 3 vstupy (SER, SRCLK, SCLK)
* 8 výstupov Q0-Q7 (potrebujeme 3 PINy na Arduine), napájame na 7segmentový displej



**Úloha 6. Zobrazovanie čísla na 7segmentovom displeji pomocou posuvného registra.**



#define DATA 2

#define SRCLK 3 //shift

#define STCLK 4//store

void setup()

{

pinMode(DATA, OUTPUT);

pinMode(SRCLK, OUTPUT);

pinMode(STCLK, OUTPUT);

}

void loop()

{

//pomocou 1 a 0 zhasínam a zapínam segmenty

byte dig[10];

dig[0] = B11111100;

dig[1] = B01100000;

dig[2] = B11011010;

dig[3] = B11110010;

dig[4] = B01100110;

dig[5] = B10110110;

dig[6] = B10111110;

dig[7] = B11100000;

dig[8] = B11111110;

dig[9] = B11110110;

byte d = dig[5]; //cislica 5 sa zobrazí na 7segmentovom displeji

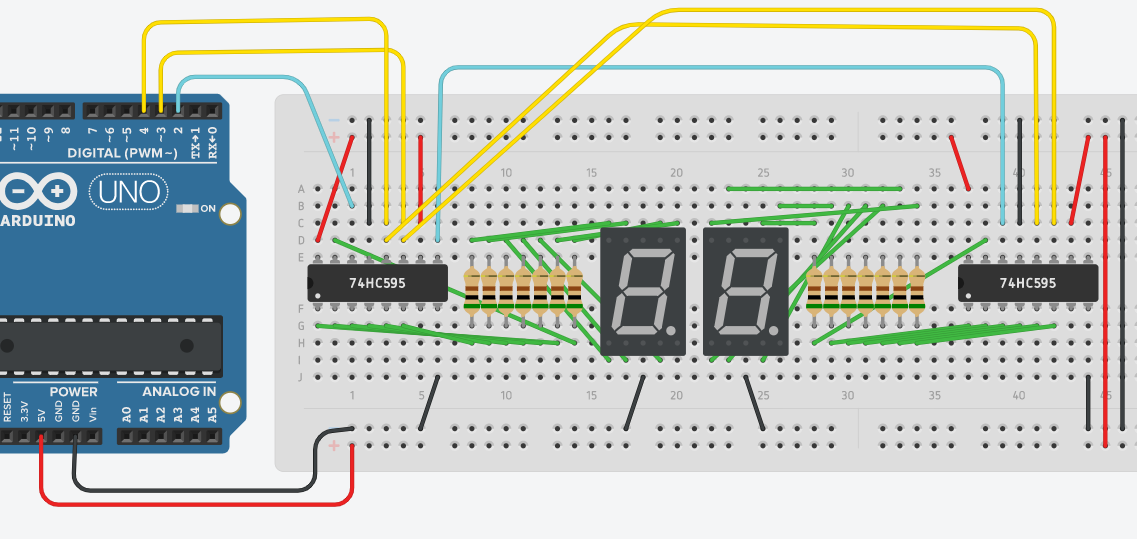
shiftOut(DATA, SRCLK, LSBFIRST, d);

digitalWrite(STCLK, HIGH);

digitalWrite(STCLK, LOW);

}

**Úloha 7. Dva 7segmentové displeje.**



#define DATA 2

#define SRCLK 3 //shift

#define STCLK 4 //store

void setup() {

pinMode(DATA, OUTPUT);

pinMode(SRCLK, OUTPUT);

pinMode(STCLK, OUTPUT);

}

void loop() {

byte dig[10]; //kody pre cislice 0-9

dig[0] = B11111100;

dig[1] = B01100000;

dig[2] = B11011010;

dig[3] = B11110010;

dig[4] = B01100110;

dig[5] = B10110110;

dig[6] = B10111110;

dig[7] = B11100000;

dig[8] = B11111110;

dig[9] = B11110110;

int d;

for(d=0; d<10; d++) {

shiftOut(DATA, SRCLK, LSBFIRST, dig[d]);

//STORE CLOCK SIGNAL

digitalWrite(STCLK, HIGH);

digitalWrite(STCLK, LOW);

delay(1000);

}

}

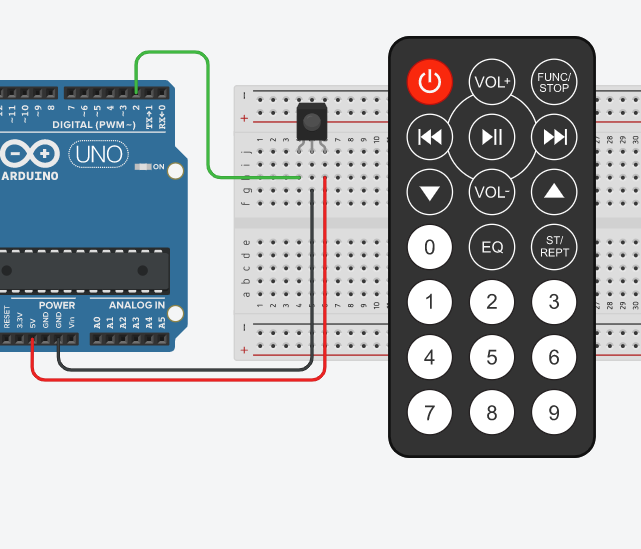
# IR ovládač

Infračervený prijímač umožňuje prijímanie signálov z IR diaľkové ovládača. Signály môžeme previesť na premenné a následne ich ďalej používať. Na odosielanie signálov je možné použiť ľubovoľné diaľkové, napr. od televízora, alebo veže a pod.

Princíp – diaľkové používajú luminenscenčné diódy infra-LED, ktoré vyžarujú žiarenie o vlnovej dĺžke cca 940nm (pri niektorých sa uvádza 950nm). Nemôžeme odosielať hodnoty svieti/nesvieti, nakoľko by pri takomto prenose dochádzalo k rušeniu napr. slnka/žiarovky. Musíme využívať moduláciu.



**Úloha 8. Vypisovanie prijímania signálu z diaľkového do Serial monitoru.**



#include <IRremote.h>

IRrecv recv(2); //číslo pinu pripojený IR senzor, inicializovaný objekt

decode\_results result;

void setup()

{

recv.enableIRIn(); //umožní prijímanie signálu

Serial.begin(9600); //inicializuje sériovú konzolu

}

void loop()

{

//ak bol prijatý signál, tak ho dekóduje

if(recv.decode(&result)) {

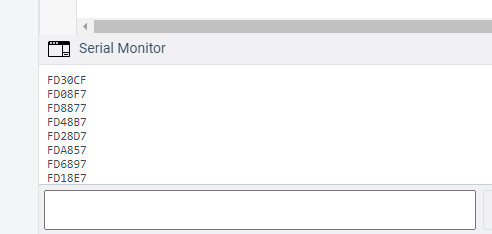
Serial.println(result.value, HEX); //vypíše kód

recv.resume();

}

}

Do seriálového monitora sa vypíšu po stlačení tlačidiel 0-9 zakódované znaky:



**Úloha 9 – modifikovanie úlohy 8, poznačíme si kódy, ktoré vysiala diaľkové po stlačení tlačidiel 0-9, vypnutie. Tieto kódy budeme prepisovať na jednociferné znaky, v prípade vypnutia na nový riadok.**

FD30CF 0

FD08F7 1

FD8877 2

FD48B7 3

FD28D7 4

FDA857 5

FD6897 6

FD18E7 7

FD9867 8

FD58A7 9

FD00FF Vypnutie

#include <IRremote.h>

IRrecv recv(2); //číslo pinu pripojený IR senzor, inicializovaný objekt

decode\_results result;

void setup()

{

recv.enableIRIn(); //umožní prijímanie signálu

Serial.begin(9600); //inicializuje sériovú konzolu

}

void loop()

{

//ak bol prijatý signál, tak ho dekóduje

if(recv.decode(&result)) {

int cislo;

//0X zadávame číslo v 16kovom formáte,za sú kódy z diaľkového

if(result.value == 0XFD30CF) cislo = 0;

if(result.value == 0XFD08F7) cislo = 1;

if(result.value == 0XFD8877) cislo = 2;

if(result.value == 0XFD48B7) cislo = 3;

if(result.value == 0XFD28D7) cislo = 4;

if(result.value == 0XFDA857) cislo = 5;

if(result.value == 0XFD6897) cislo = 6;

if(result.value == 0XFD18E7) cislo = 7;

if(result.value == 0XFD9867) cislo = 8;

if(result.value == 0XFD58A7) cislo = 9;

if(result.value == 0XFD00FF) {

Serial.println();

} else {

Serial.print(cislo);

}

recv.resume();

}

}

Ďalšie možnosti riešenia nájdete v predpripravených úlohách v Arduino IDE je v časti Example IRsendDemo