**ČÍSELNÉ SÚSTAVY**

* Tvorba čísel v 2, 10, 16 sústave
* Násobky 2 na N
* Prevod z 2 do 10
* Prevod z 10 do 2
* Prevod z 2 do 16
* Prevod z 16 do 2
* Aritmetický súčin v binárnej sústave
* Aritmetický súčet v binárnej sústave
* Logický súčet
* Logický súčin

**Práca s registrami v jazyku C/C++/C#/...**

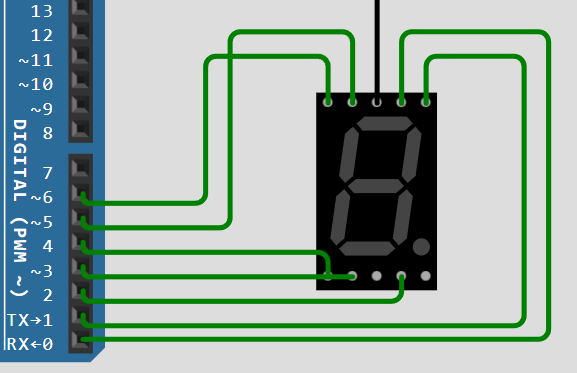
* Nastavenie celého registra na 0 a 1
* Nastavenie viacerých 1 v registry
* Nastavenie viacerých 0 v registry
* Nastavenie jedného registra do log. 1
* Nastavenie jedného registra do log. 0

**Registre pre I/O (vstup/výstup)**

* Register DDRx
* Register PORTx

**Príklad na digitálny výstup cez register**

* Blikanie LED diódy
* Semafór
* Postupné rozsvecovanie a zhasínanie bargraphu
* Efekt loadingu na 7-segmentovom displeji
* Postupné zobrazenie čísel 0-9 na 7-segmentovom displeji
* Zobrazenie čísla na 7-segmentovom displeji v závislosti od zadania čísla cez Serial



byte cisla[10] = // pole pre 10 bajtov (10x znakov)

{

  0b00111111, // bajt reprezentujuci kombinaciu H/L na pinoch pre znaku 0

  0b00000110, // znak 1

  0b01011011, // znak 2

  0b01001111,

  0b01100110,

  0b01101101,

  0b01111101,

  0b00000111,

  0b01111111,

  0b01101111

};

void setup()

{

  DDRD |= 0b01111111; // prvych 7 pinov ako vystup

}

void loop()

{

  for(int i=0;i<10;i++) // cyklus, ktory prejde 10-krat

  {

    PORTD = cisla[i]; // nastav kombinaciu i-teho znaku v poli

    delay(255); // pockah 255 ms

  }

}

**Práca s Proteus\_ISIS a Arduino\_IDE resp. TinkerCAD**

* Oboznámenie sa so softvérom pre simuláciu dosiek Arduino a ich programovania

**Práca s registrami v jazyku C/C++/C#/...**

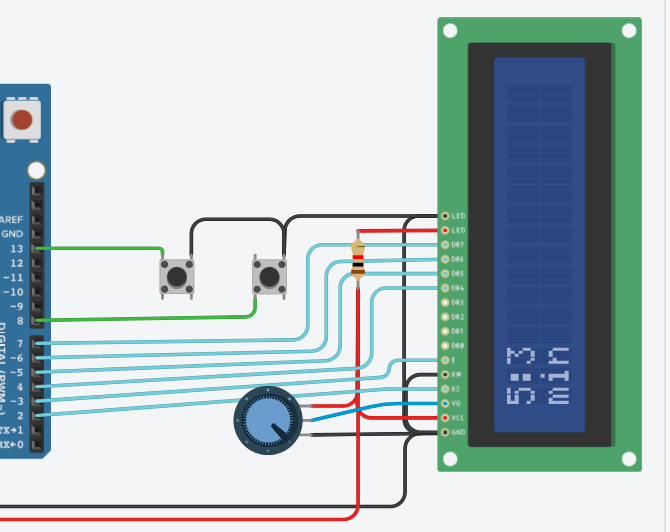
* Načítanie hodnoty z registra
* Otestovanie log.0 / log.1 v registry

**Registre pre I/O (vstup/výstup)**

* Register PINx

**Príklad na digitálny vstup cez register**

* Prepínanie farieb na semafóre
* Postupne sa zvyšujúce číslo na LCD displeji po stlačení tlačidla
* Ovládanie dvoch motorov (tankový posun) cez Serial
* Ovládanie dvoch motorov (tankový posun) cez 4x tlačidlá
* Práca s prepínačom (4 pinový DIP prepínač) – zmena X/O v rohoch displeja
* Prepínanie znakov X/O v rohoch pomocou 4x DIP prepínačov
* Detekcia pohybu pomocou PIR senzora (LCD displej + RGB LED dióda)
* Nastavovanie času na LCD v tvare MM:SS



#include <LiquidCrystal.h>

LiquidCrystal lcd(2,3,4,5,6,7);

int minuty = 0;

int sekundy = 0;

int ktore = 0; // 0=m 1=s

void setup()

{

DDRB &= ~(1<<5);

PORTB |= 1<<5;

DDRB &= ~(1<<0);

PORTB |= 1<<0;

lcd.begin(16,2);

lcd.print(String(minuty) + ":" + String(sekundy));

lcd.setCursor(0, 1);

lcd.print("min");

}

void loop()

{

if( (PINB&(1<<0)) == 0 )

{

if(ktore == 0) // zmena na prvom riadku

{

minuty++;

lcd.home();

lcd.print(String(minuty) + ":" + String(sekundy));

}

else

{

sekundy++;

lcd.home();

lcd.print(String(minuty) + ":" + String(sekundy));

}

delay(500);

}

if( (PINB&(1<<5)) == 0 ) // prepnutie druhého riadku

{

if(ktore == 0)

{

ktore = 1;

lcd.setCursor(0, 1);

lcd.print("sek");

}

else

{

ktore = 0;

lcd.setCursor(0, 1);

lcd.print("min");

}

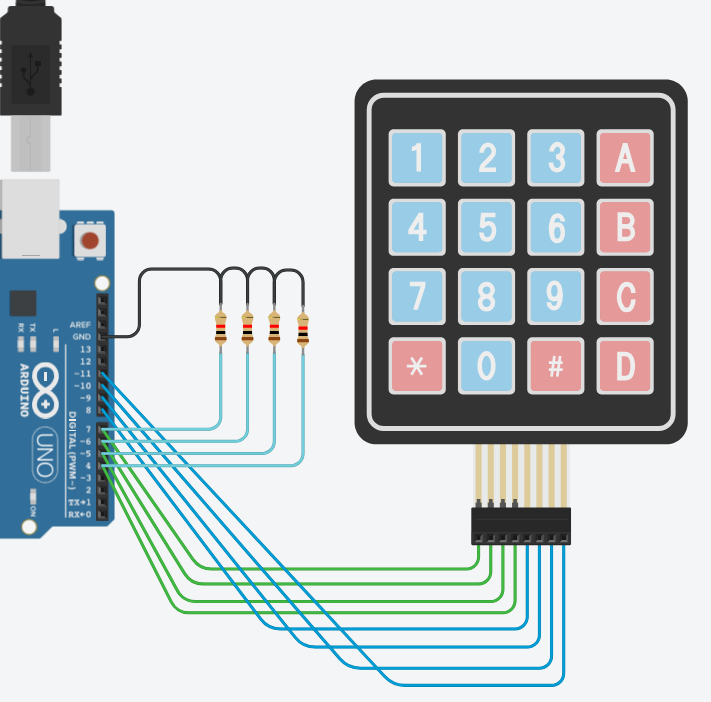
delay(500);

}

}

**Príklad na keypad cez registre a cez knižnicu**

* Zachytenie stlačenia tlačidiel pomocou registrov a výpis do Serialu
* Zachytenie stlačenia tlačidiel pomocou knižnice a výpis na 7-seg. displej



void setup()

{

Serial.begin(9600);

DDRB |= 0b00001111;

DDRD &= 0b00001111;

}

void loop()

{

PORTB = 0b00000001;

if( (PIND&(1<<7)) != 0 )

{

Serial.println("1");

}

if( (PIND&(1<<6)) != 0 )

{

Serial.println("4");

}

if( (PIND&(1<<5)) != 0 )

{

Serial.println("7");

}

if( (PIND&(1<<4)) != 0 )

{

Serial.println("\*");

}

delay(5);

PORTB = 0b00000010;

if( (PIND&(1<<7)) != 0 )

{

Serial.println("2");

}

if( (PIND&(1<<6)) != 0 )

{

Serial.println("5");

}

if( (PIND&(1<<5)) != 0 )

{

Serial.println("8");

}

if( (PIND&(1<<4)) != 0 )

{

Serial.println("0");

}

delay(5);

delay(100);

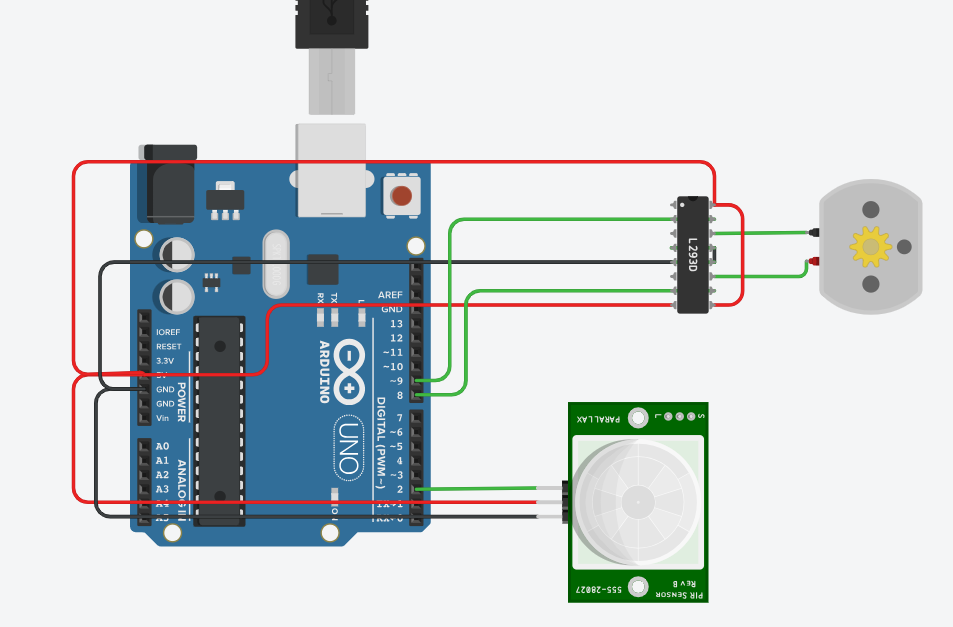
}

**Registre pre vonkajšie prerušenie**

* Registre: EICRA, EIMSK, SREG
* Vektor prerušenia a obsluha prerušenia ISR (interrupt service routine)

**Príklad na vonkajšie prerušenie**

* Zvyšovanie čísla na Serial popri blikaní LED
* Zvyšovanie a znižovanie čísla v rozmedzí 0-9
* Prepínanie semafóru medzi klasickým režimom a blikaním – dve prerušenia
* Pozastavenie motora pri narušení Safe-zóny (PIR) – mód any change



bool tociSa = true;

void setup()

{

DDRD &= ~(1<<2); // pir ako vstup

DDRB |= 1<<0;

DDRB |= 1<<1;

SREG |= 1<<7;

EIMSK |= 1<<INT0;

EICRA &= ~(1<<ISC01);

EICRA |= 1<<ISC00;

}

void loop()

{

if(tociSa == true)

{

PORTB |= 1<<0;

PORTB &= ~(1<<1);

}

else

{

PORTB &= ~(1<<0);

PORTB &= ~(1<<1);

}

}

ISR(INT0\_vect)

{

if( (PIND&(1<<2)) == 0 )

tociSa = true;

else

tociSa = false;

}

**Registre pre pin-change prerušenie**

* Registre: