Introducere în Arduino - Laborator 2

Scopul lucrării

- **Înțelegerea operațiilor de intrare/ieșire**
- **Înțelegerea modulării semnalelor în lățime**
- **Înțelegerea convertorului analog-digital**
- **Dezvoltarea unor aplicații folosind conceptele învățate**
- **Challenge (fun)**

Recapitulare

Arduino este un *microcontroller*, în esență, un sistem de calcul compact format din porturi de intrare/ieșire, memorie(Ram și Rom) și unitatea centrală de prelucrare(CPU). Arduino a fost proiectat pentru a facilita interacțiunea cu mediul înconjurător prin senzori și actuatori, dar are și capacitatea de a executa funcții intensive d.p.d.v. al puterii de calcul.

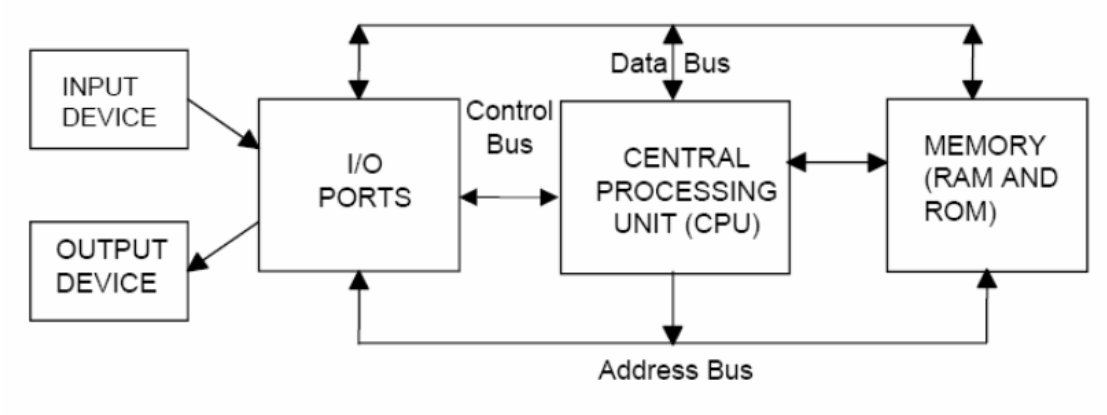


Figura 1 - Arduino schema bloc

Structura porturilor de Intrare/Ieșire

- Porturi I/O Arduino Uno: **B, C, D.**
- Fiecare bit/pin dintr-un port poate fi configurat ca intrare sau ieșire cu ajutorul registrului DDRx(reg. de direcție).
- Registrul PORTx, pentru scrierea valorilor dorite pe porturi.
- PINx este folosit la citirea starii pinilor.
- Diode de protecție împotriva electricității statice.
- Rezistența de ”pull-up”.
- Pentru fiecare port se alocă trei locații în spațiul de adrese: **PORTx, DDRx, PINx**.

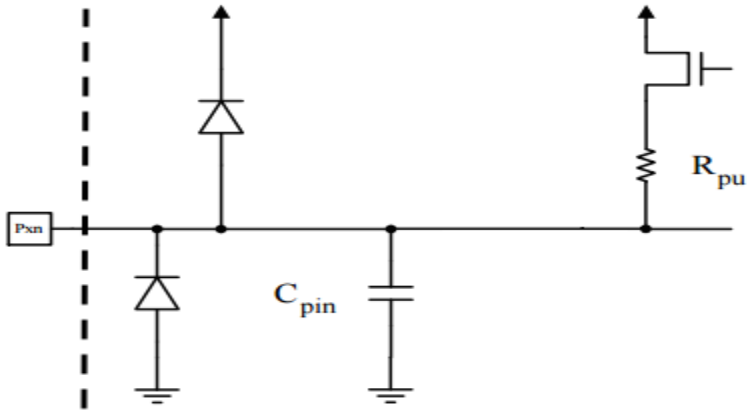


Figura 2 - Diode de protecție

Prin intermediul funcției pinMode(pin, INPUT), scriem conținutul registrului DDR pentru portul în care se află pinul.

Registru DDR(Roșu) controlează bufferul cu 3 stări care conectează bistabilul PORT(Verde) la pin. Poarta ȘI(Albastru) controlează rezistența de pull-up, activă doar cand bitul din DDR este setat ca INPUT(0), bitul din registrul PORT este 1, iar PUD(Pull-up disable) este 0.

Prin citirea registrului PIN(mov), veți obține mereu starea curentă a pinului(sincronizată cu semnalul de ceas - nu există un bistabil lacăt care să salveze valoarea). Prin scrierea registrului PIN veți comuta bitul registrului PORT, prin trimiterea valorii bistabilului PORT, înapoi la intrarea acestuia(Galben)

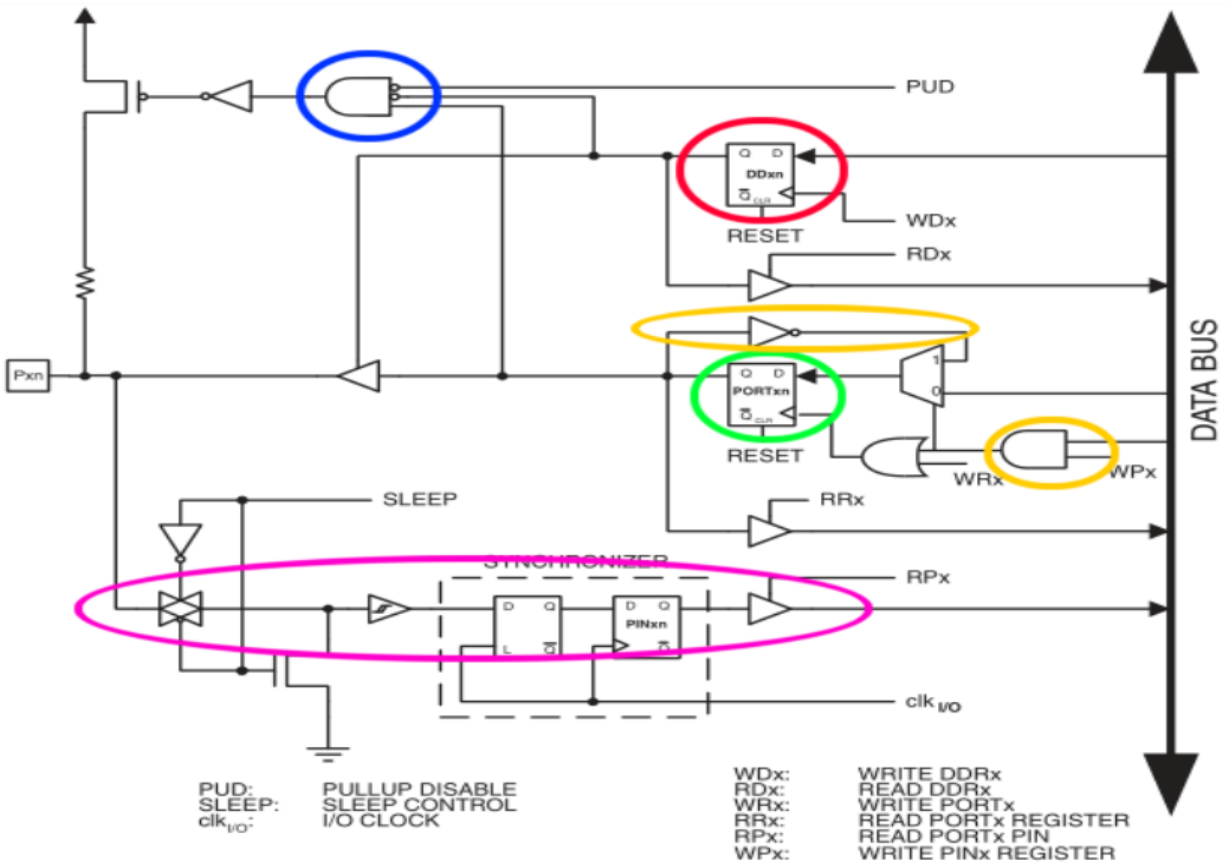


Figura 3 - Moduri de configurare a unui pin

Rezistența de pull-up

_____Rezistența de pull-up(sau de pull-down) este folosită în circuitele logice pentru a asigura un nivel logic bine definit unui pin sub orice circumstanțe. Circuitele logice digitale au trei stări logice: HIGH(1 logic), LOW(0 logic) și HIGH_IMPEDANCE(”floating”). Dacă un pin se află în starea de înaltă impendanță, micrcontrollerul nu poate citi corect valorile acelui pin, drept urmare, este folosită rezistența de pull-up pentru a asigura 5V constant pinului.
Similar rezistența de pull-down este folosită pentru a asigura 0V constant pinului.

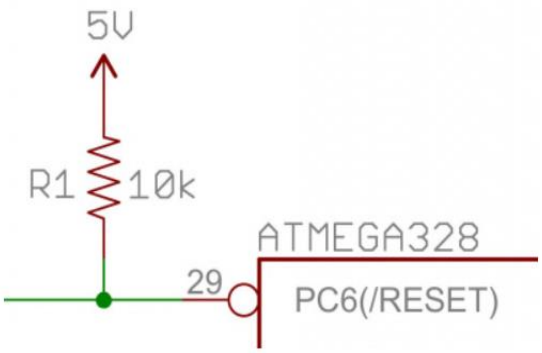


Figura 4 - Pull-up resistor

Pulse Width Modulation

PWM este o tehnică cu ajutorul căreia putem modifica media tensiunii(”average voltage”) într-un circuit, astfel putem controla circuite analogice prin ieșirile digitale ale unui microcontroller. Pentru a înțelege PWM trebuie să înțelegem Duty Cycle.

Duty Cycle reprezintă fracțiunea dintr-o perioadă în care semnalul este activ(1 logic). În mod implicit Duty Cycle pentru un semnal dreptunghiular este 50%, asta înseamnă că jumătate de perioadă semnalul este activ. Folosind drept exemplu un semnal dreptunghiular cu amplitudinea 10V, Duty Cycle de 50%, înseamnă că media tensiunii este 5V(deoarece pentru cealaltă jumătate de perioada semnalul este inactiv => (0.5 * 10 + 0.5*0 = 5V).

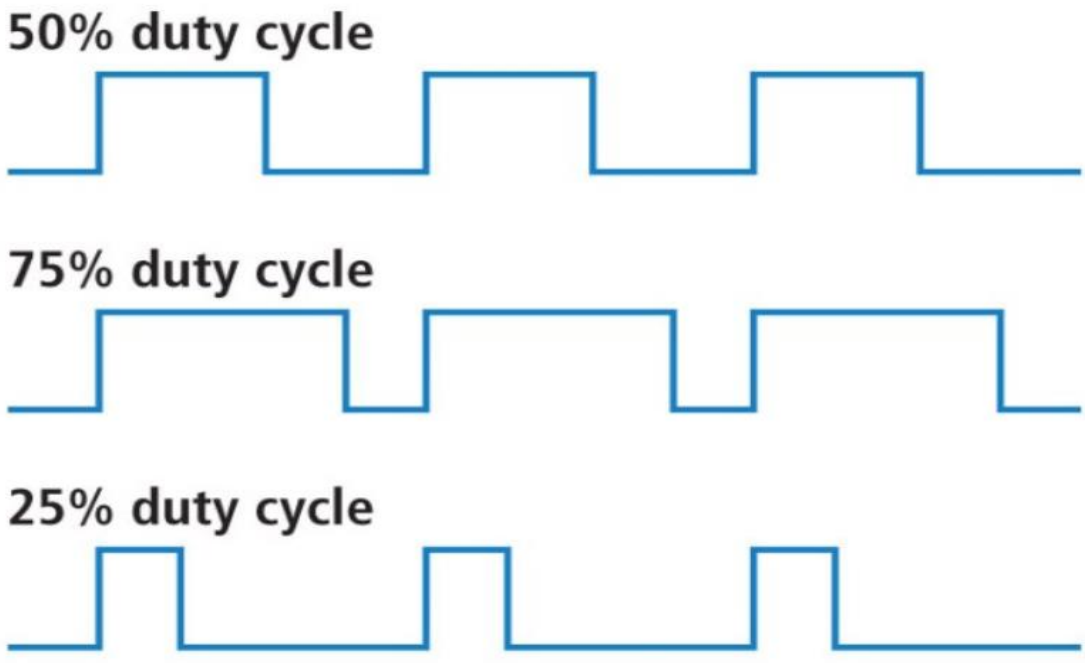


Figura 5 - Duty cycle

Convertor analog-digital

Convertește o tensiune analogică de intrare într-o valoare digitală pe 10 biți. Tensiunea la un anumit moment dat este păstrată cu ajutorul componentului de sample and hold(Roșu) apoi trimisă comparatorului(Galben). Registrul de aproximări succesive(SAR - Verde), generează o valoare inițială setând bitul cel mai semnificativ. Acestă valoare este convertită în tensiune analogică cu ajutorul unui convertor digital-analogic(DAC - mov) după formula: (SARVal * URef)/MaxSARVal. Valoarea obținută este de asemenea trimisă comparatorului. Comparatorul decide dacă tensiunea citită inițial este mai mare decât valoarea obținută cu ajutorul SAR, în

caz afirmativ comparatorul va seta ieșirea, altfel ieșirea va avea valoarea 0. Dacă ieșirea comparatorului are valoarea 1, SAR setează urmatorul bit, altfel valoarea bitului curent devine 0, iar următorul devine 1
.

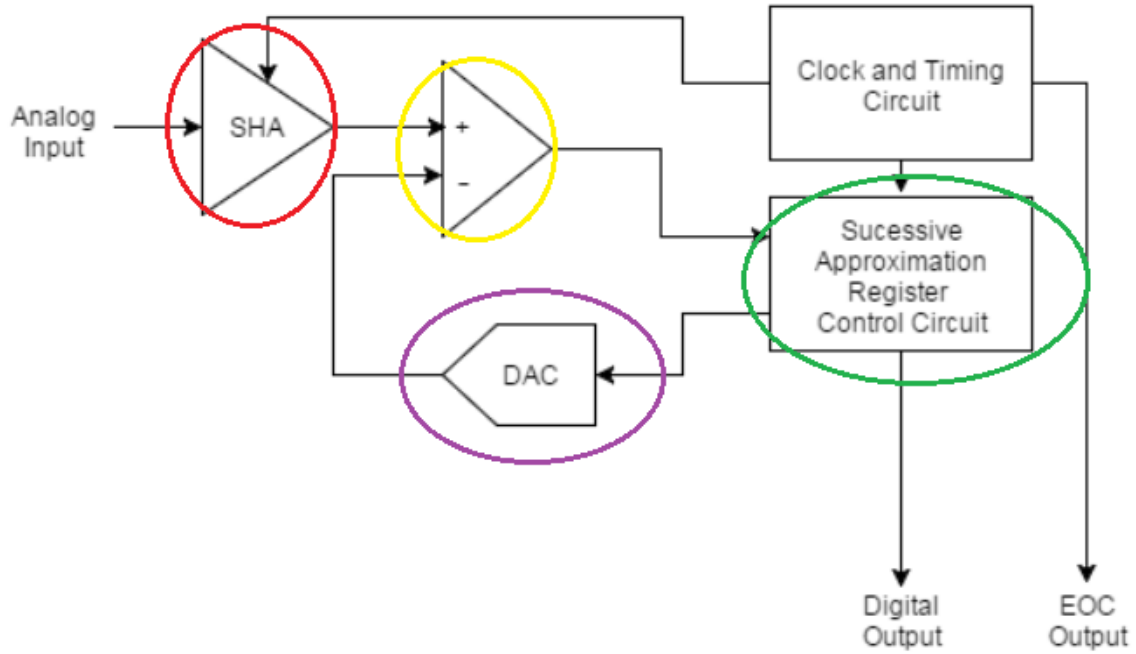


Figura 6 - ADC

Întrebări premergătoare aplicațiilor practice

- Cum funcționează un breadboard?
- Aflați valoarea rezistenței necesară pentru funcționarea corectă a unui L.E.D
- Ce este un divizor de tensiune?
- Care este rolul functiilor pinMode, digitalWrite(pin, value), digitalRead(pin)?

În cadrul circuitelor electronice, un breadboard este un panou electronic pentru conectarea elementelor de circuit fără sudură. Acesta este utilizat pentru realizarea circuitelor temporare și a prototipurilor și nu necesită nici un fel de lipire.

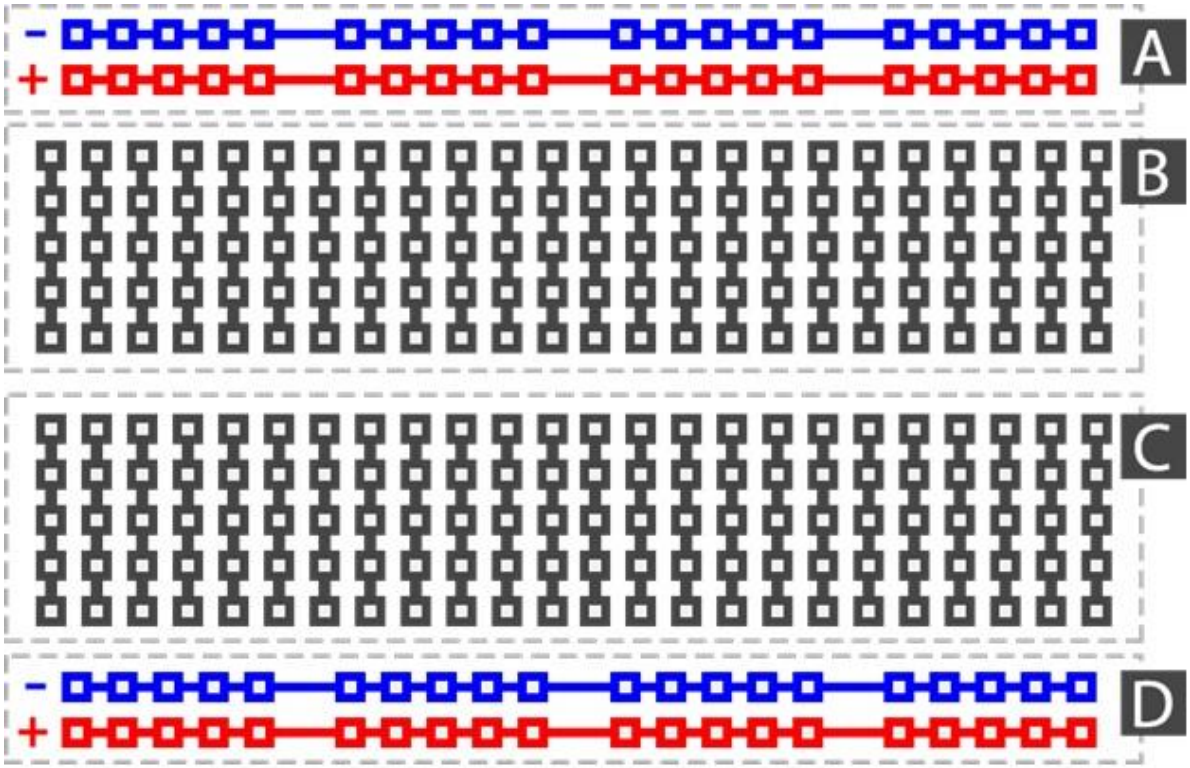


Figura 7 - Schemă breadboard

Aplicații

Blink and fade

Încărcați exemplul din laboratorul anterior, Blink (File - Examples - Basics - Blink) și realizați schema electrică de mai jos.

După ce ați realizat schema electrică, alimentați placa, încărcați codul și observați comportamentul ledului(Similar cu cel din laboratorul trecut dar nu folosim ledul încorporat pe placa). Acum încărcați Fade (File - Examples - Basics - Fade). Modificați schema electrică conectând ledul la pinul 9 nu la 13(deoarece folosim un pin capabil de PWM). Înțelegeți codul înainte de a-l încărca după care puteți analiza comportamentul ledului.

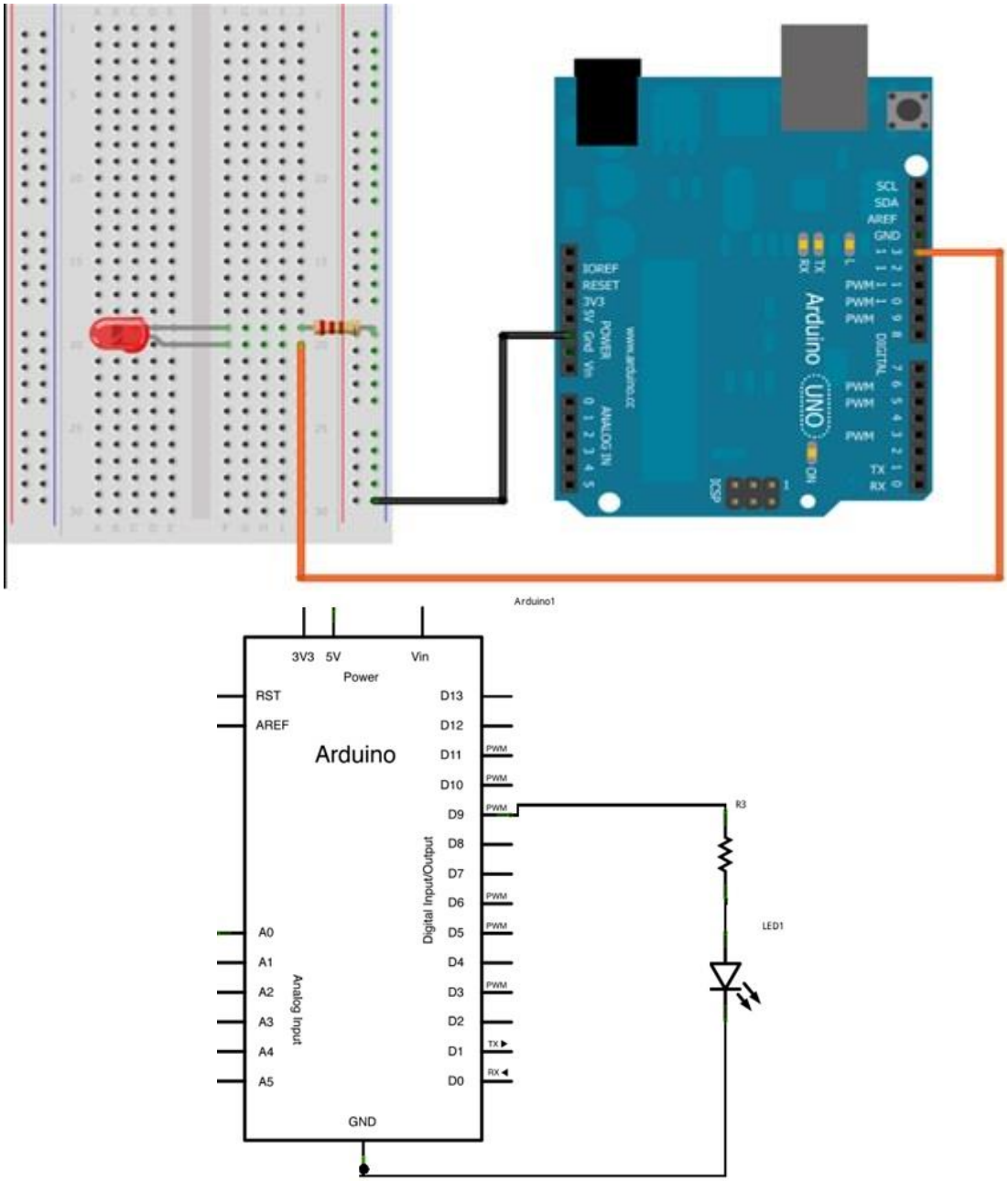


Figura 8 - Configurație pentru led

Switch

Vom folosi un buton, pentru a controla starea unui led. Realizați schema electrică din imagine, iar apoi încărcați codul. Ce tip de rezistență este folosit? Este nevoie de rezistența externă neapărat?

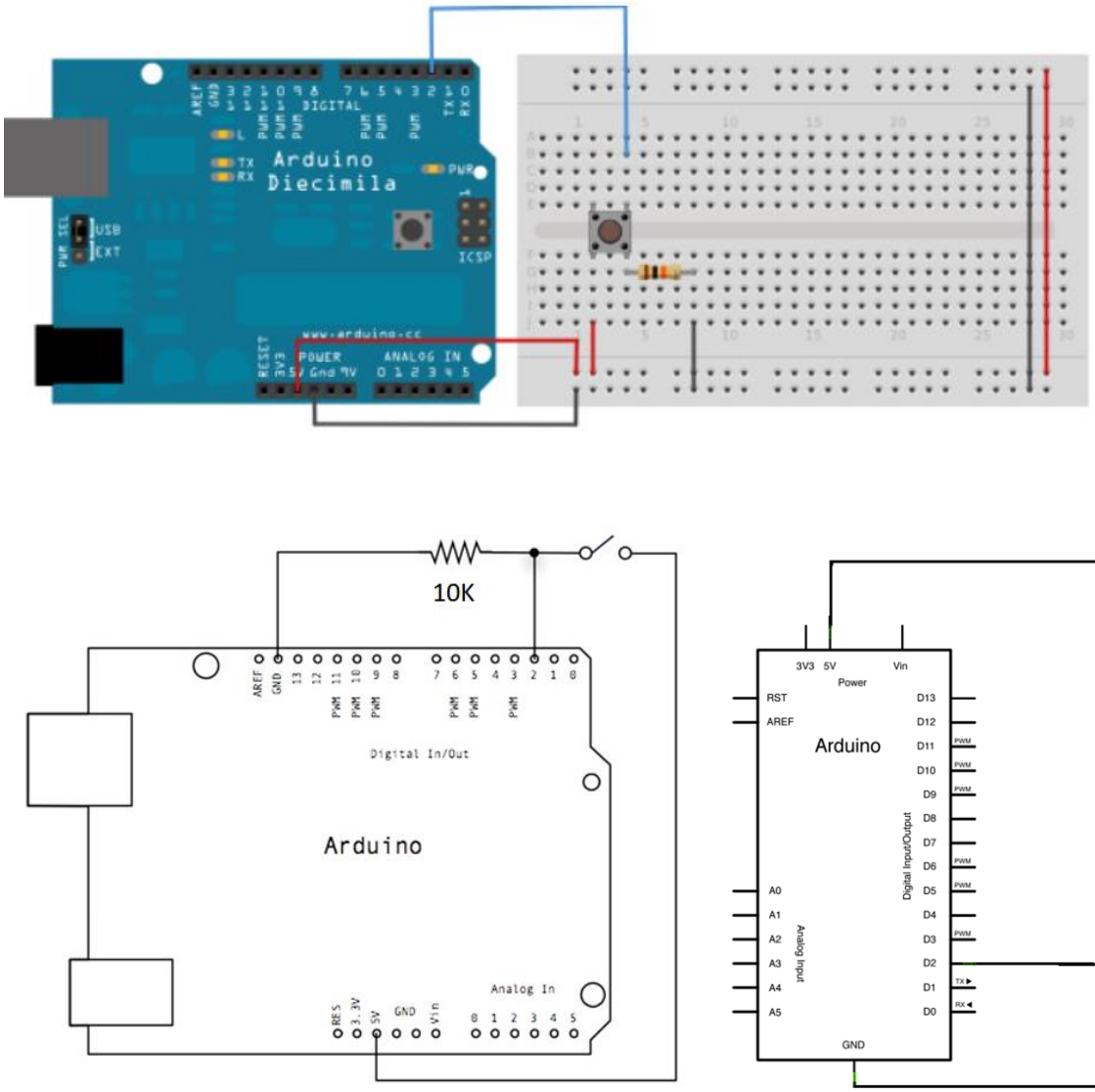


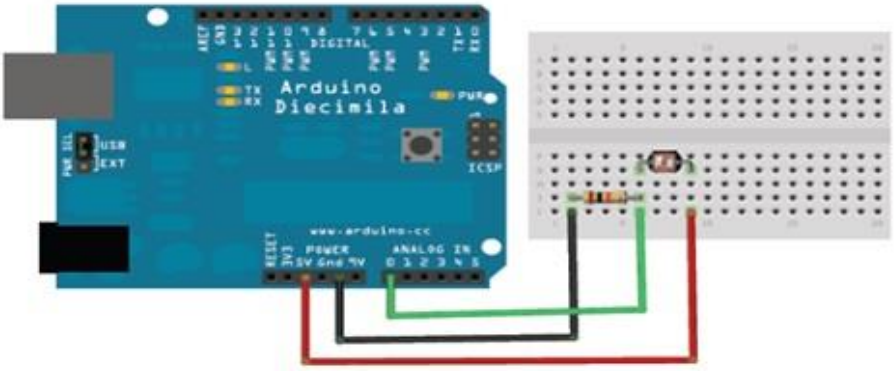
Figura 9 - Configurație pentru switch

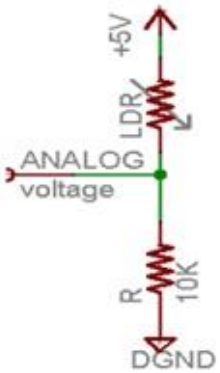
```
// constants won't change. They're used here to set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
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);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Figura 10 - Cod switch

Photocell

Vom folosi o celulă sensibilă la lumină împreună cu o rezistență pe post de divizor de tensiune. Celula își modifică rezistența în funcție de intensitatea luminii ambientale, astfel obținem un divizor de tensiune cu o rezistență variabilă. Deoarece tensiunea furnizată de divizor variază, o vom citi cu ajutorul pinilor analogici (folosind funcția analogRead()), iar valoarea digitală(obținută de la convertorul analog-digital) va fi folosită pentru a exprima intensitatea luminoasa citită de celulă. Realizați schema apoi citiți și înțelegeți codul înainte de a-l încărca.





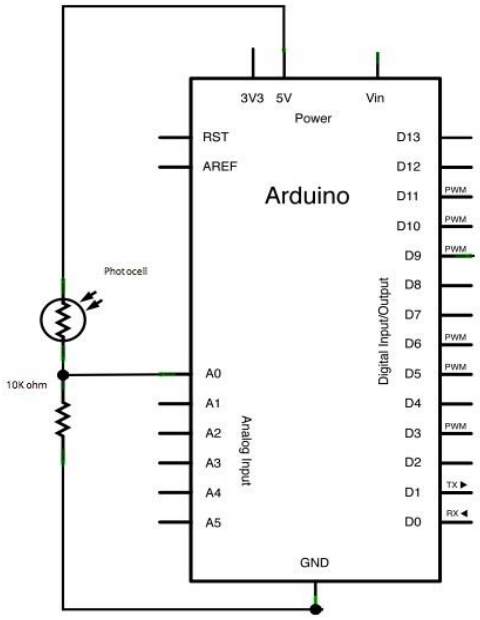


Figura 11 - Configurație photocell

```
int photocellPin = 0;     // the cell and 10K pulldown are connected to a0
int photocellReading;     // the analog reading from the analog resistor divider

void setup(void) {
  // We'll send debugging information via the Serial monitor
  Serial.begin(9600);
}

void loop(void) {
  photocellReading = analogRead(photocellPin);

  Serial.print("Analog reading = ");
  Serial.print(photocellReading);     // the raw analog reading

  // We'll have a few threshholds, qualitatively determined
  if (photocellReading < 10) {
    Serial.println(" - Dark");
  } else if (photocellReading < 200) {
    Serial.println(" - Dim");
  } else if (photocellReading < 500) {
    Serial.println(" - Light");
  } else if (photocellReading < 800) {
    Serial.println(" - Bright");
  } else {
    Serial.println(" - Very bright");
  }
  delay(1000);
}
```

Figura 12 - Cod photocell

Challenge - Theremin

Un theremin este un instrument muzical electronic, controlat de mișcarea mâinilor artistului (vezi Youtube). Pentru a realiza acest instrument vom folosi o fotocelulă astfel. Vom citi valorile divizorului de tensiune, similar problemei anterioare, iar valoarea citită la un moment dat va fi trimisă unui buzzer și va reprezenta frecvența sunetului. Pentru a comanda buzzerul vom folosi funcția tone(pin, frequency). Buzzerul are două terminale (Vcc si GND) si are nevoie de o rezistență de 100 ohmi.

Intrebări postmergătoare laboratorului

- Cum funcționează un convertol analog-digital?
- Care este rolul funcției analogRead()?
- Când se află un pin în starea ”floating”?
- Pentru ce este utilizată rezistența de pull-down?