

PROJECT DOCUMENT

Digital Challenge

Cover page

PROJECT NUMBER

2024-2-RS01-KA210-SCH-000271542

Садржај

Увод у Arduino: Први кораци у програмирању микроконтролера.....	3
Развојна окружења: Arduino IDE, Tinkercad и Circuit Designer.....	6
Arduino IDE.....	6
Tinkercad и Circuit Designer.....	6
Сензори и актуатори који се користе у наредним пројектима.....	6
Вежба 1: Учитавање првог програма — Blink.....	7
Вежба 2: Повезивање спољашње LED диоде.....	8
.....	8
Увод у Processing: Визуелизација података у реалном времену за Arduino пројекте.....	12
Развојно окружење Processing (PDE).....	16
Вежба 1: Први Processing програм.....	22
Вежба 2: Примање података са Arduino.....	23
Вежба 3: Обрада више вредности (CSV формат).....	25
Вежба 4: Интеракција са Processingом — слање команди Arduino.....	27
Пројекат 1: Температура и влажност — DHT11.....	35
Вежба 1: Повезивање са DHT11 сензором.....	36
Wiring.....	36
Вежба 2: Визуелизација података у Processing окружењу.....	38
Пројекат 2: Сензор осветљености — Фотоотпорник (LDR).....	41
Вежба 1: Повезивање кола и Arduino програм.....	42
Вежба 2: Визуелизација у Processing окружењу.....	43
Project 3: Sonar — Ultrasonic Sensor HC-SR04.....	46
Вежба 1: Мерење растојања и приказ у Processing-у.....	47
Вежба 2: Контрола SG90 серво мотора из Processing-а.....	49
Компоненте.....	49
Вежба 3: Ултразвучни сензор контролише серво мотор.....	52
Вежба 4: Направи ултразвучни радар (сонар).....	54
Пројекат 4: Arduino мерач отпорности са приказом вредности у Processing-у.....	57
Пројект 5: Аутоматски систем за наводњавање.....	61
.....	62
Вежба 1: Прикупљање података и визуелизација.....	63
Вежба 2: Управљање пумпом за воду из Processing-а.....	65
.....	66
Вежба 3: Комплетно аутоматски систем за наводњавање.....	67
Хакатон и примена стечених знања.....	73
Педагошке и методичке основе.....	81

Увод у Arduino: Први кораци у програмирању микроконтролера

Опис

У овој наставној јединици ученици се упознају са развојном плочом Arduino Uno, основним елементима микроконтролерског система и првим корацима у раду са електронским колима. Посебан нагласак стављен је на повезивање софтвера и хардвера, јер управо та веза представља основу сваког STEM пројекта – од почетне идеје, преко симулације, до израде функционалног уређаја.

Наставна јединица је припремљена у складу са принципима Erasmus+ STEM пројекта и подстиче истраживачко учење, практичну примену знања и разумевање начина на који једноставан програм може управљати стварним уређајем.

Овај модул представља уводну основу за све наредне пројекте са сензорима и визуелизацијом података у окружењу Processing у оквиру радионица „Дигитални изазов“

Циљеви

Ученици ће:

- разумети шта је Arduino и чему служи микроконтролерска развојна плоча;
- препознати основне делове плоче Arduino Uno;
- научити да користе breadboard плочу и отпорнике у једноставном електронском колу;
- повезати програмски код са физичким повезивањем електронских компоненти;
- отпремити свој први програм помоћу окружења Arduino IDE;
- развити навику да пре практичне реализације провере шему повезивања и исправност кола.

Резултати учења

Након реализације ове наставне јединице ученици ће бити у стању да:

- објасне основну намену Arduino платформе;
- препознају најважније делове плоче Arduino Uno;
- самостално саставе једноставно електронско коло на breadboard плочи;
- покрену и модификују основни Arduino програм;
- користе функције pinMode(), digitalWrite() и delay();
- повежу симулацију електронског кола са његовом реалном реализацијом.

Потребна опрема

Component	Quantity	Purpose
Arduino Uno	1	Главна развојна плоча
USB кабл	1	Повезивање са рачунаром и учитавање програма
Breadboard плоча	1	Брзо повезивање компоненти без лемљења
LED диода	1–2	Визуелна индикација рада програма
220 Ω отпорник	1–2	Ограничење струје кроз LED диоду

Component	Quantity	Purpose
Jumper проводници	~4	Повезивање компоненти
Arduino IDE	1	Писање и учитавање програма
Tinkercad или Circuit Designer	1	Симулација и провера електронског кола

Теоријске основе

Шта је Arduino?

Arduino је хардверско-софтверска платформа заснована на микроконтролеру. У пракси то значи да мала развојна плоча може да читава сигнале са сензора, обрађује прикупљене податке и управља различитим уређајима као што су LED диоде, мотори, релеји, дисплеји и други електронски склопови.

Захваљујући једноставном повезивању компоненти, великом броју доступних примера и богатој колекцији библиотека, Arduino је изузетно погодан за почетнике, али и за брзу израду прототипова у настави и пројектним активностима..

💡 Arduino можемо посматрати као „мозак“ пројекта – он прима информације са улаза (сензори, тастери) и управља излазима (LED диоде, мотори, дисплеји и други уређаји).

Зашто Arduino Uno?

Arduino Uno је најчешћи избор за почетне вежбе јер је поуздан, добро документован и једноставан за употребу. Поседује довољан број улазно-излазних прикључака за већину школских експеримената, а истовремено није сувише сложен за ученике који се први пут сусрећу са микроконтролерима.

Након усвајања основних знања на плочи Arduino Uno, прелазак на друге моделе, као што су Arduino Nano, Arduino Mega или ESP32, постаје знатно једноставнији.

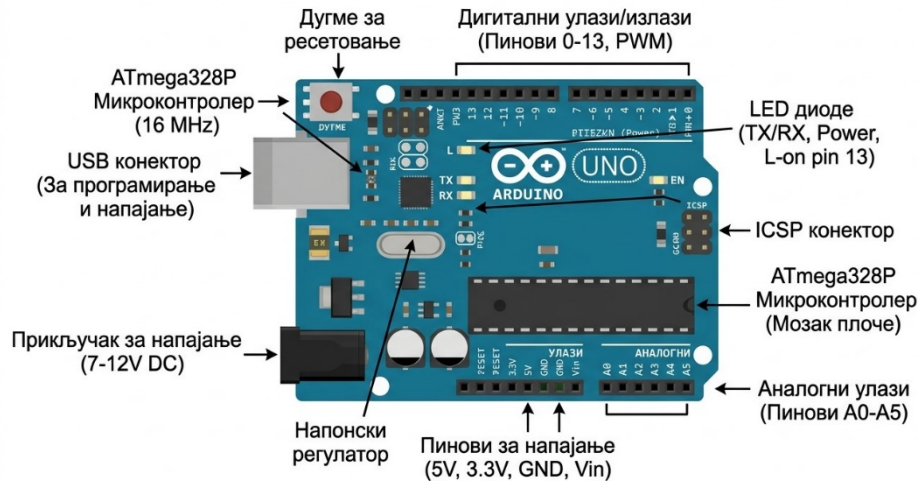
Основни делови микроконтролера Arduino Uno

Пре првог повезивања електронског кола важно је да ученици препознају најважније делове плоче:

- USB конектор – служи за повезивање са рачунаром и учитавање програма;
- дигитални пинови (0–13) – користе се за дигиталне улазне и излазне сигнале;
- аналогни пинови (A0–A5) – користе се за читавање аналогних вредности са сензора;
- GND – прикључак за масу (референтни потенцијал кола);
- 5V и 3.3V – прикључци за напајање компоненти;
- уграђена LED диода – повезана је на пин 13 и погодна је за прве тестове;
- RESET дугме – поново покреће програм на микроконтролеру;
- микроконтролер ATmega328P – главни процесор плоче који извршава програм.

✓ **Савет:** Пре повезивања било које компоненте увек пронађите GND и пинове за напајање како бисте избегли неправилно повезивање и могуће оштећење компоненти.

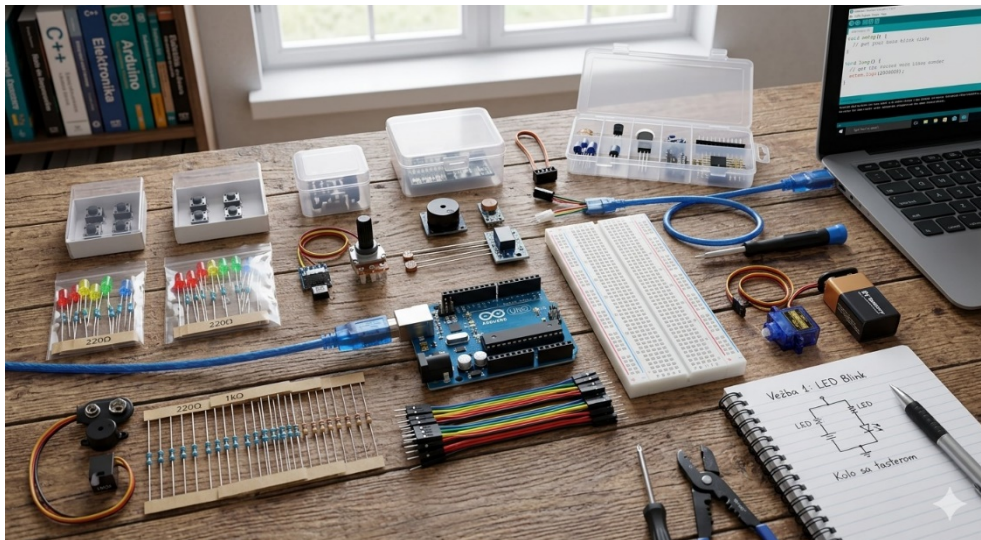
ОСНОВНИ ДЕЛОВИ ARDUINO UNO ПЛОЧЕ



Основни комплет

За прве вежбе користи се основни комплет за учење електронике, који обично садржи Arduino плочу, USB кабл, breadboard плочу, проводнике за повезивање, отпорнике, LED диоде, тастер, сензоре и друге мање електронске компоненте. Овакав комплет омогућава извођење великог броја занимљивих експеримената већ од самог почетка рада.

Када ученици науче да препознају компоненте и правилно их повезују, могу самостално да реализују читав низ вежби без страха од погрешног повезивања.



Breadboard плоча, отпорници и LED диода

Breadboard плоча служи за брзо повезивање електронских компоненти без лемљења. Она омогућава једноставне измене кола, испробавање различитих идеја и проверу исправности повезивања пре израде трајног решења.

Отпорник има улогу да ограничи струју у електричном колу. При раду са LED диодама употреба отпорника је неопходна како би се спречило њихово оштећење услед превелике струје.

Приликом повезивања LED диоде важно је знати која ножица представља аноду (+), а која катоду (-), као и правилно користити GND прикључак.

💡 Анода LED диоде (дужа ножица) повезује се са сигналним пином преко отпорника, док се катода (краћа ножица) повезује на GND.

Развојна окружења: Arduino IDE, Tinkercad и Circuit Designer

Програми за Arduino најчешће се пишу у окружењу Arduino IDE. У њему се програмски код уноси, проверава и учитава у микроконтролер.

Tinkercad и Circuit Designer су корисни алати за планирање електронских кола и њихову почетну проверу, јер омогућавају симулацију рада пре реалног повезивања компоненти.

Arduino IDE

Arduino IDE је званично програмско окружење за Arduino развојне плоче. У њему се пише програмски код, проверава исправност програма и врши његово учитавање у микроконтролер.

Ово окружење је једноставно за употребу и веома погодно за почетнике.

Arduino IDE омогућава:

- писање и уређивање програма (sketch);
- учитавање програма на Arduino плочу;
- праћење рада програма помоћу алата Serial Monitor;
- додавање библиотека за рад са сензорима и другим компонентама.

Софтвер је бесплатан и може се преузети са званичне Arduino интернет странице. Након инсталације потребно је одабрати одговарајући тип плоче и COM порт, а затим учитати програм на уређај.

Tinkercad и Circuit Designer

електронских кола пре њихове реализације. Ово је посебно корисно у настави и током првих вежби са Arduino платформом јер омогућава уочавање и исправљање грешака пре рада са стварним компонентама.

✓ Препоручена пракса: Најпре симулирајте коло у окружењу Tinkercad, а тек након успешне провере реализујте га на breadboard плочи

Сензори и актуатори који се користе у наредним пројектима

У наредним вежбама и пројектима ученици ће радити са различитим сензорима и актуаторима који омогућавају прикупљање података из окружења и управљање електронским системима.

Sensor / Actuator	Type	Used In
DHT11	Сензор температуре и влажности ваздуха	Пројакат 1
LDR (фоторезистор)	Сензор осветљености	Пројакат 2
HC-SR04	Ултразвучни сензор растојања	Пројакат 3
Relay module	Електронски прекидач (актуатор)	Пројакат 5
Сензор влажности земљишта	Капацитивни сензор влажности земљишта	Пројакат 5
SG90 Серво мотор	Актуатор за управљање положајем	Пројакат 3

Вежба 1: Учитавање првог програма — Blink

Ова вежба служи за проверу да ли је Arduino плоча исправно повезана са рачунаром, да ли је изабран одговарајући COM порт и да ли се програм успешно учитава у микроконтролер.

Arduino код

```
const int ledPin = 13;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```

Објашњење кода

Команда	Објашњење
const int ledPin = 13;	Одређује се пин на коме је LED диода.
setup()	Покреће се једном при стартовању програма.
pinMode(ledPin, OUTPUT);	Пин се дефинише као излазни output.
loop()	Понавља се све док је Ардуино укључен.
digitalWrite(ledPin, HIGH/LOW);	Укључује LED ако је (HIGH) или искључује ако је (LOW).
delay(1000);	Паузира се извршавање програма једну секунду (1000 милисекунди).

Слање програма на Arduino

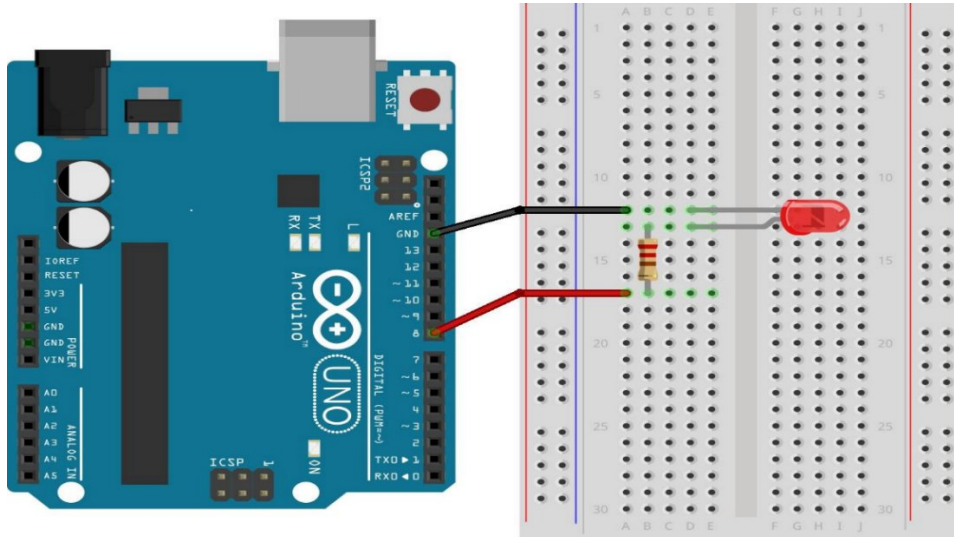
- Покрените Arduino IDE.
- У менију Tools → Board изаберите Arduino Uno.
- У менију Tools → Port изаберите одговарајући COM порт.
- Кликните на дугме Upload (икона стрелице →).
- Након успешног учитавања програма, уграђена LED диода на плочи треба да почне да трепери у интервалу од једне секунде.

✓ Ако LED диода трепери, програм је успешно уčitан, а Arduino плоча је исправно повезана и спремна за рад

Вежба 2: Повезивање спољашње LED диоде

У овој вежби ученици прелазе са уграђене LED диоде на спољашњу LED диоду и по први пут самостално састављају једноставно електронско коло. На овом примеру јасно се уочава веза између шеме повезивања, реалног кола и програмског кода који управља његовим радом..

Повезивање



Слика 3: Реално повезивање на breadboard-у

Arduino Uno	Component	Connection
Pin 8	LED анода (+)	Преко отпорника 220 Ω
GND	LED катода (-)	Директно на масу

Arduino код

```
const int ledPin = 8;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  digitalWrite(ledPin, HIGH);
  delay(500);
  digitalWrite(ledPin, LOW);
  delay(500);
}
```

Шта ради овај програм?

- укључује LED диоду на пину 8,
- чека пола секунде (500 ms),

- искључује LED диоду,
- понавља процес укључивања и искључивања.

Задаци за ученике

Ниво	Задатак	Циљ
● Основни	Променити вредност функције delay()	Разумети утицај трајања паузе
● Средњи	Повезати LED диоду на други пин	Увежбати додељивање пинова у програму
● Напредни	Додати тастер за управљање LED диодом	Упознати функције INPUT_PULLUP и digitalRead()
● Изазов	Направити семафор са три LED диоде	Рад са више излаза и временском логиком

Савети и честе грешке

Обратите пажњу на следеће:

- уек прво пронађите GND и пинове за напајање пре повезивања компоненти;
- никада не повезујте LED диоду без отпорника за ограничење струје;
- пре учитавања програма проверите да ли је изабран исправан COM порт;
- ако коло не ради, најпре проверите физичке везе, а затим програмски код;
- почните са једноставним колима пре него што пређете на сложеније пројекте.

💡 . Систематичан приступ раду: Шема → Симулација (Tinkercad) → Реално повезивање → Учитавање програма → Тестирање

Практична примена

Овакав приступ учењу користи се у:

- образовним радионицама и STEM активностима;
- развоју прототипова за паметне куће;
- првим IoT пројектима и повезаним уређајима;
- метеоролошким и мерним системима;
- роботским и аутоматизованим решењима;
- интерактивним светлосним инсталацијама;
- maker пројектима.

Међупредметне везе

Предмет	Веза	Пример
Физика	Мерење и разумевање физичких појава	Напон, струја, отпор, сензори
Информатика	Програмирање и алгоритми	setup(), loop(), услови, петље
Техника и технологија	Електроника и практичан рад	Повезивање компоненти, склапање кола
Математика	Логичко размишљање и анализа	Временски интервали, поређење вредности, бинарна стања
Ликовна култура и дизајн	Креативна примена технологије	Светлосни ефекти, интерактивни прикази

DigComp компетенције

Ова наставна јединица развија следеће компетенције према оквиру DigComp 2.2:

DigComp област	Развијена компетенција
1. Информациона и дигитална писменост/ Писменост у раду са подацима	Разумевање дигиталних сигнала и података са сензора
2. Комуникација и сарадња	Представљање резултата и рад у пару или групи
3. Креирање дигиталних садржаја	Писање и измена Arduino програма (sketch)
4. Безбедност	Безбедна употреба електронских компоненти и одговарајућих напонских нивоа
5. Решавање проблема	Систематично отклањање грешака у колу и програму

Посебно се развијају:

- дигитална писменост и основе рачунарског размишљања;
- решавање проблема кроз систематично отклањање грешака;
- основне програмерске вештине кроз рад са микроконтролером;
- разумевање везе између хардвера и софтвера;
- тимски рад и сарадња у STEM окружењу.

Идеје за проширење пројекта

Ученици који успешно реализују обе вежбе могу проширити пројекат на следеће начине:

- додавањем сервомотора и програмирањем његовог кретања;
- повезивањем сензора осветљености LDR ради аутоматског управљања осветљеношћу LED диоде;
- коришћењем алата Serial Monitor за приказ порука и вредности са сензора;
- симулацијом електронског кола у окружењу Tinkercad пре реалне реализације;
- преласком на ESP32 у каснијој фази пројекта ради успостављања WiFi комуникације;
- додавањем тастера и програмирањем укључивања и искључивања LED диоде;
- израдом семафорског система са три LED диоде и временски дефинисаном секвенцом рада.

Проширење	Ниво сложености	Нова знања и вештине
Поруке у Serial Monitor-у	● Основни	Serial.begin(), Serial.println()
Управљање помоћу тастера	● Средњи	digitalRead(), INPUT_PULLUP, логика стања
Управљање осветљењем помоћу LDR сензора	● Средњи	analogRead(), map(), калибрација сензора
Управљање сервомотором	● Напредни	Servo библиотека, PWM сигнали

Проширење	Ниво сложености	Нова знања и вештине
Управљање LED диодом преко WiFi мреже (ESP32)	● Изазов	WiFi библиотека, web сервер, IoT концепти

Закључак

Ова наставна јединица представља први корак у раду са Arduino платформом и поставља основу за све наредне STEM активности у оквиру радионица „Дигитални изазов“. Кроз вежбу Blink и повезивање спољашње LED диоде ученици уочавају како програмски код, електронске компоненте и физичко коло функционишу као јединствен систем.

Знања и вештине стечени у овој наставној јединици — читање шеме повезивања, писање функција `setup()` и `loop()`, употреба функција `digitalWrite()` и `delay()`, као и рад на breadboard плочи — непосредно се примењују у свим наредним пројектима, од мерења температуре до сонарских и аутоматизованих система.

Литература и наредни кораци

Као наставак ове наставне јединице препоручују се следећи пројекти из серијала „Дигитални изазов“:

Следећи пројекат	Тема	Нови концепти
Пројекат 1	Температура и влажност – DHT11	Библиотеке, серијска комуникација, Processing
Пројекат 2	Сензор осветљености – LDR	Аналогни улаз, напонски делилац, стубичасти дијаграм
Пројекат 3	Сонар – HC-SR04	Мерење растојања, сервомотор, радарски приказ
Пројекат 4	Мерач отпорности	Омов закон, мерење помоћу напонског делиоца
Пројекат 5	Аутоматско наводњавање	Релеј, управљање пумпом, логика прагова

✓ Препоручени редослед рада: Blink вежбе → Пројекат 1 (DHT11) → наставак рада кроз свих пет пројеката редом

Introduction to Arduino | Arduino + Processing | Digital Challenge | 2024-2-RS01-KA210-SCH-000271542

Увод у Processing: Визуелизација података у реалном времену за Arduino пројекте

Опис

Processing је бесплатно програмско окружење отвореног кода намењено креирању визуелних приказа, анимација и интерактивних апликација. У контексту Arduino пројеката, Processing има улогу визуелног интерфејса: прима податке које Arduino шаље путем серијске комуникације и приказује их у реалном времену у облику текста, стубичастих приказа, графикона, боја, геометријских облика или анимација.

Ова наставна јединица припрема ученике за свих пет пројеката у оквиру радионица „Дигитални изазов“, у којима се Processing користи за визуелизацију података са сензора – од температуре и влажности ваздуха до растојања, интензитета осветљености, отпорности и влажности земљишта.

Посебан нагласак стављен је на учење кроз истраживање и практично разумевање начина на који се подаци из физичког окружења преносе и приказују у визуелном облику.

Циљеви

Ученици ће:

- разумети шта је Processing и по чему се разликује од окружења Arduino IDE;
- инсталирати и подесити Processing окружење;
- разумети структуру Processing програма (setup() и draw());
- научити како се примају подаци са Arduino плоче путем серијске комуникације;
- приказивати вредности са сензора помоћу текста, графичких облика и стубичастих приказа у боји;
- користити функцију serialEvent() за ефикасну обраду података;
- направити комплетан систем за визуелизацију података у реалном времену повезан са Arduino сензором..

Резултати учења

Након реализације ове наставне јединице ученици ће бити у стању да:

- објасне улогу окружења Processing у систему Arduino + рачунар;
- напишу и покрену једноставан Processing програм;
- користе библиотеку Serial за пријем података са Arduino плоче;
- обрађују вредности раздвојене зарезима које шаље Arduino;
- приказују вредности са сензора помоћу функција text(), rect(), ellipse() и fill();
- реагују на пристигле податке коришћењем функције serialEvent();
- израде визуелни приказ података у реалном времену за најмање један сензорски пројекат.

Потребан софтвер и опрема

Компонента	Верзија / Детаљи	Намена
Processing IDE	4.x (најновија верзија) – processing.org	Писање и покретање визуелизација
Arduino IDE	2.x (најновија верзија) – arduino.cc	Програмирање Arduinoa
Arduino Uno	Развојни систем	Слање података путем серијске комуникације

Компонента	Верзија / Детаљи	Намена
USB cable	Стандардни A-B или A-micro	Повезивање Arduinoa са рачунаром
Сензори	DHT11, LDR, HC-SR04, etc.	Прикупљање података за визуелизацију

✓ Processing је бесплатан софтвер и доступан је за оперативне системе Windows, macOS и Linux. Може се преузети са адресе: processing.org/download

Теоријске основе

Шта је Processing?

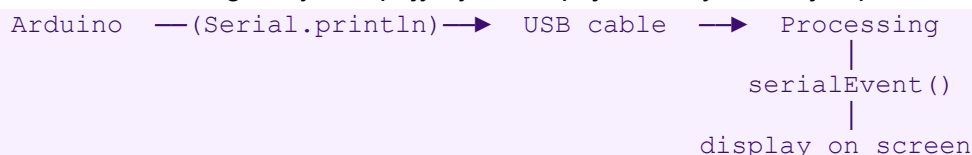
Processing је програмски језик и развојно окружење засновано на програмском језику Java, развијено првенствено за визуелни дизајн и интерактивне апликације. Настао је 2001. године у оквиру MIT Media Lab-a и данас се широко користи у области дигиталне уметности, визуелизације података, образовања и креативног програмирања.

У Arduino екосистему Processing има улогу визуелног интерфејса. Иако не може директно да управља хардвером, веома је погодан за пријем података и њихово претварање у прегледне визуелне приказе

Карактеристика	Arduino IDE	Processing
Основна улога	Управљање хардвером (сензори, мотори)	Визуелизација података (графика, графикони)
Извршавање програма	Микроконтролер (Arduino плоча)	Рачунар (Windows, macOS, Linux)
Издаз	Сигнали ка компонентама	Графички приказ на екрану
Ток података	Шаље податке путем серијске комуникације	Прима податке путем серијске комуникације
Основа програмског језика	C / C++	Java, Python, Javascript
Структура програма	setup() + loop()	setup() + draw()

Како Arduino и Processing сарађују?

Arduino и Processing комуницирају путем серијске комуникације преко USB кабла:



Ток података у свих пет пројеката „Дигиталног изазова“ заснива се на истом принципу:

- сензор мери физичку величину (температуру, растојање, осветљеност и сл.);
- Arduino обрађује измерену вредност и шаље је помоћу функције Serial.println();
- Processing прима текстуалну поруку, обрађује је и претвара у бројчану вредност;
- Processing приказује податке на екрану помоћу текста, стубичастих приказа, боја или графичких облика.

💡 Оба програма морају користити исту брзину серијске комуникације (baud rate). На пример: Arduino: `Serial.begin(9600)`; Processing: `new Serial(this, port, 9600)`.

Структура Processing програма

Сваки Processing програм (који се назива *sketch*) садржи две основне функције – потпуно исто као и Arduino програм:

```
void setup() {
  // Runs ONCE at the start
  // Set window size, initialize serial port
}

void draw() {
  // Runs CONTINUOUSLY – like loop() in Arduino
  // Draw shapes, text, update the display
}
```

Функција	Када се извршава	Примена
setup()	Једном када се програм покрене	size(), подешавање серијске комуникације, почетне променљиве
draw()	Непрекидно (~60 fps)	background(), text(), rect(), графички облици
serialEvent()	Кад пристигну нови подаци са сериал порта	Читање и обрада података са Arduina
mousePressed()	Приликом клика на тастер миша	Дугмад корисничког интерфејса, интеракције
keyPressed()	Приликом притиска на тастер тастатуре	Слање команди на Arduino

Библиотека Serial

Processing комуницира са Arduinoм помоћу уграђене библиотеке Serial. Основни кораци:

```
import processing.serial.*; // 1. Import the library

Serial myPort; // 2. Declare serial port object

void setup() {
  size(400, 300);
  // 3. List available ports (helpful for finding the right one)
  printArray(Serial.list());
  // 4. Open the port – index [0] is usually the first available
  myPort = new Serial(this, Serial.list()[0], 9600);
}
```

✓ Ако `Serial.list()[0]` није исправан порт, проверите индекс. Покрените `printArray(Serial.list())` како бисте приказали све доступне портове.

⚠ Затворите Serial Monitor у Arduino IDE пре покретања Processing програма — оба програма не могу истовремено користити исти серијски порт.

Основне функције за цртање у Processing окружењу

Функција	Опис	Пример
size(w, h)	Подешава ширину и висину прозора	size(600, 400)
background(r,g,b)	Боји позадину	background(255) — white
fill(r,g,b)	Боји графичке облике	fill(255, 0, 0) — red
stroke(r,g,b)	Боји ивице графичких објеката	stroke(0) — black
text(str, x, y)	Исписује текст на задатом месту	text("Hello", 50, 100)
textSize(n)	Подешавање величине слова	textSize(20)
rect(x, y, w, h)	Црта правоугаоник	rect(50, 100, 200, 30)
ellipse(x, y, w, h)	Црта елипсу/круг	ellipse(200, 200, 50, 50)
line(x1,y1,x2,y2)	Црта линију	line(0, 0, 200, 200)
map(val,lo1,hi1,lo2,hi2)	Пресликава вредност у нови опсег	map(temp, 0, 100, 0, 300)
nf(n, d1, d2)	Форматира број као текст	nf(3.14159, 1, 2) → "3.14"

Боје у Processing окружењу

Processing користи RGB вредности (0–255) за све функције које раде са бојама:

```
fill(255, 0, 0);      // Red
fill(0, 255, 0);     // Green
fill(0, 0, 255);     // Blue
fill(255, 255, 0);   // Yellow
fill(0);              // Black (shorthand)
fill(255);            // White (shorthand)
fill(128, 0, 128);   // Purple

// With transparency (alpha channel 0-255):
fill(255, 0, 0, 150); // Semi-transparent red
```

Функција map()— основна функција за визуелизацију података

Функција map() пресликава вредност из једног опсега у други. Ово је веома важно приликом приказивања података са сензора помоћу графичких елемената, јер вредности које сензори шаљу (0–1023) треба прилагодити величини екрана:

```
// Sensor value 0-1023 mapped to bar width 0-300 pixels:
float barWidth = map(sensorValue, 0, 1023, 0, 300);
rect(50, 100, barWidth, 30);

// Temperature 0-50°C mapped to color red component:
float r = map(temperature, 0, 50, 0, 255);
fill(r, 0, 0);
```

Развојно окружење Processing (PDE)

Развојно окружење Processing (PDE) обухвата текстуални едитор, компајлер и прозор за приказ. Оно омогућава креирање софтвера у оквиру пажљиво осмишљеног скупа ограничења.

PDE се састоји од једноставног текстуалног едитора за писање кода, поља за поруке, текстуалне конзоле, картица за управљање фајловима, траке са алатима за често коришћене радње и низа менија.

Погледајте линк: <https://processing.org/environment>

Компонента PDE-а	Намена
Текстуални едитор	Писање и уређивање кода Processing скице
Поље за поруке	Повратна информација приликом чувања, извоза и приказа грешака
Конзола	Текстуални излаз из скица — резултати print() и println()
Трака са алатима	Дугмад Run и Stop за брзу контролу скице
Картице	Управљање више фајлова унутар једне скице
Менији	File, Edit, Sketch, Debug, Tools, Help

💡 У Processing-у, компјутерски програм се назива скица (sketch). Скице се чувају у Sketchbook-у — фолдеру на вашем рачунару.

Дугмад на траци са алатима

Дугме	Радња
▶ Run	Компајлира код и отвара нови прозор за приказ (Java режим)
■ Stop	Прекида покренуту скицу

Мени File

Команда	Опис
New	Креира нову скицу именовану по тренутном датуму (формат: sketch_YYMMDDa)
Open...	Отвара скицу у новом прозору
Open Recent	Бира из недавно затворених скица
Sketchbook...	Приказује листу свих скица у sketchbook-у
Examples...	Прегледање уграђених и додатних примера кода
Close	Затвара тренутни прозор скице
Save	Чува отворену скицу у тренутном стању
Save As...	Чување под другим именом (не замењује претходну верзију)
Export Application...	Извоз као самосталан извршни фајл и отварање излазног фолдера
Preferences...	Подешавања Processing-а (на macOS-у: мени Processing)
Quit	Излазак из Processing-а и затварање свих прозора (на macOS-у: мени Processing)

Мени Edit

Команда	Опис
Undo	Поништава последњу команду или унос
Redo	Поништава последњу Undo радњу
Cut	Уклања и копира означени текст у клипборд
Copy	Копира означени текст у клипборд
Copy as HTML	Копира код форматиран као HTML (изглед идентичан као у PDE-у)
Paste	Убацује садржај клипборда на позицију курсора
Select All	Означава сав текст у тренутно отвореном фајлу
Auto Format	Форматира код у читљивији распоред (раније звано Beautify)
Comment / Uncomment	Коментарише или уклања коментар означеног текста
Increase Indent	Увлачи означени текст за два размака
Decrease Indent	Уклања два размака из увлачења означеног текста
Find...	Проналажење и опционо замена текстуалног низа

Команда	Опис
Find Next	Проналажење следећег појављивања траженог низа
Find Previous	Проналажење претходног појављивања траженог низа

Мени Sketch

Команда	Опис
Run	Компајлира, отвара прозор за приказ и покреће скицу
Present	Покреће скицу преко целог екрана на једнобојној позадини (Esc за излазак)
Tweak	Покреће са уживо уређивањем боја и вредности променљивих (скица мора прво бити сачувана)
Stop	Зауставља тренутно покренуту скицу
Import Library...	Додаје import наредбе; изаберите Add Libraries... за инсталацију нових
Show Sketch Folder	Отвара фолдер који садржи фајлове тренутне скице
Add File...	Додаје слику, фонт или медија фајл у data фолдер скице

Мени Debug

Команда	Опис
Enable Debugger	Активира дебагер (дугме Run се мења у Debug)
Continue	Наставља извршавање кода до следеће прекидне тачке
Step	Напредује једну линију по једну
Step Into	Улази у позив кориснички дефинисане функције
Step Out	Изази из функције назад у позивајући део
Toggle Breakpoint	Додаје или уклања прекидну тачку (број линије се замењује ознаком <>)

💡 Дебагер се активира преко Debug → Enable Debugger, или кликом на иконицу лептира у горњем десном углу PDE-а.

Мени Tools

Команда	Опис
Archive Sketch	Чува .zip копију тренутне скице у истом фолдеру
Color Selector...	Бирање боја и копирање њихових HSB, RGB или Hex вредности
Create Font...	Конвертује системске фонтове у Processing VLBW формат за коришћење у скицама
Theme Selector	Бирање или креирање прилагођених PDE тема боја
Movie Maker	Креира QuickTime филм од низа извезених слика
Manage Tools...	Отвара Contribution Manager за прегледање и инсталацију алата

Подешавања (Preferences)

PDE је веома подешавав. Најчешћа подешавања се мењају у прозору Preferences (мени File на Windows/Linux, мени Processing на macOS-у):

Подешавање	Опис
Sketchbook location	Поставља фолдер који се користи као Sketchbook
Language	Бирање језика менија (потребан рестарт)
Editor font size	Поставља величину фонта у едитору кода
Console font size	Поставља величину фонта у конзоли
Background color when Presenting	Боја која се користи приликом покретања скице у Present режиму
Continuously check for errors	Укључује/искључује детекцију грешака у реалном времену
Code completion with Ctrl-Space	Укључује/искључује аутоматско довршавање кода
Suggest import statements	Аутоматски предлаже библиотеке када се препозна њихов код
Increase maximum available memory	Додељује више RAM меморије скицама које користе велике медија фајлове
Run sketches on display	Бира који монитор се користи за прозор скице (вишеструки монитори)

Скице и Sketchbook

Сви Processing пројекти се називају скице. Свака скица има сопствени фолдер — главни фајл има исто име као фолдер, са екстензијом .pde (Processing Development Environment).

```
Назив скице:      MyProject
Фолдер:           MyProject/
Главни фајл:      MyProject/MyProject.pde
Медија фајлови:  MyProject/data/ ← слике, фонтови, звукови иду овде
```

Скице се по подразумеваним подешавањима чувају у Sketchbook-у. Да бисте пронашли његову локацију:

- Идите на File → Preferences (или Processing → Preferences на macOS-у)
- Проверите поље Sketchbook location на врху

✓ Додајте слике, фонтове или звучне фајлове преко Sketch → Add File... — аутоматски се смештају у data/ фолдер.

Рендерери

Processing има три уграђена рендерера екрана који се бирају кроз функцију size():

Рендерер	Опис	Најбоље за
Default (без аргумента)	Стандардни 2D рендерер	Већину 2D скица и приказе сензора
P2D	Бржи али мање прецизан 2D рендерер (OpenGL)	2D графику велике брзине
P3D	3D геометрија са камером, осветљењем, материјалима (OpenGL)	3D визуелизације, радар
FX2D	Висококвалитетан 2D рендерер (одвојена JavaFX библиотека)	Велике или високорезолуцијске приказе

```
// Подразумевани 2D рендерер:
void setup() { size(400, 300); }

// P3D рендерер за 3D графику:
void setup() { size(400, 300, P3D); }
```

Координатни систем

Processing користи координатни систем где се исходиште (0, 0) налази у горњем левом углу. X расте удесно; Y расте наниже.

```
// У прозору 320×240:
// (0, 0)      = горњи леви угао
// (320, 240)  = доњи десни угао
// (319, 239) = последњи видљиви пиксел (пиксели се цртају десно и испод
// координате)

// У P3D-у, z=0 је на површини екрана; негативно z иде у дубину сцене.
```

💡 Запамтите: Y расте наниже. rect(50, 100, 200, 30) црта почевши 100 пиксела од ВРХА — не од дна.

Стилови програмирања

Processing подржава два нивоа сложености скице:

Стил	Опис	Користити када
Статична скица	Без <code>setup()</code> или <code>draw()</code> — код се извршава једном и зауставља	Учење основа: координате, облици, боје
Активна скица	Користи петљу <code>setup()</code> + <code>draw()</code> за анимацију и интеракцију	Подаци сензора, приказ у реалном времену, интерактивни UI

Пример статичне скице

```
// Извршава се једном — црта жути правоугаоник:
size(200, 200);
background(255);
noStroke();
fill(255, 204, 0);
rect(30, 20, 50, 50);
```

Пример активне скице — прати миша

```
void setup() {
  size(200, 200);
  noStroke();
  fill(0, 102, 153, 204);
}

void draw() {
  background(255);
  rect(width - mouseX, height - mouseY, 50, 50);
  rect(mouseX, mouseY, 50, 50);
}
```

Картице и вишеструки фајлови

За веће скице, код се може поделити на више картица. Кликните на дугме са стрелицом десно од картица да бисте креирали, преименовали или обрисали картице.

Када се покрене програм са више картица, све картице се компајлирају заједно. Класе у другим картицама постају унутрашње класе — не могу имати статичке променљиве. За потпуно одвојене фајлове у Java стилу, додајте картицу са екстензијом `.java`.

- ✓ Картице су намењене напредним корисницима. За пројекте Digital Challenge, довољна је скица са једним фајлом.

Додавање библиотеке, алата и режима

Шта додати	Како додати
Библиотека	Sketch → Import Library → Add Library... → изабрати → Install
Алат	Tools → Manage Tools... → изабрати → Install
Режим	Мени Mode (горе десно) → Add Mode... → изабрати → Install
Примери	File → Examples... → дугме Add Examples → изабрати → Install

За пројекте Digital Challenge, инсталирајте Serial библиотеку (уграђена) и проверите да ли је доступна:

```
// На врху сваке пројектне скице:
import processing.serial.*;
```

Командно-линијски интерфејс (CLI)

Processing укључује CLI за покретање скица из терминала — користан за аутоматизацију и headless окружења:

ОС	Команда
Windows	processing --help more
Linux	processing --help
macOS	cd /Applications/Processing.app/Contents/MacOS затим ./Processing --help

Вежба 1: Први Processing програм

Пре повезивања са Arduino плочом, ученици се упознају са Processing окружењем кроз израду једноставног анимираног програма који црта облике и текст — без потребе за хардвером.

Циљ

Нацртати прозор са правоугаоником у боји, текстом и кругом који временом мења боју.

Processing код

```
int frameCount = 0;

void setup() {
  size(500, 300);
  textAlign(CENTER, CENTER);
}

void draw() {
  background(30); // Dark gray background
  frameCount++;

  // Animated color using sin wave
  float r = 127 + 127 * sin(frameCount * 0.05);
  float g = 127 + 127 * sin(frameCount * 0.03);

  // Colored rectangle
  fill(r, g, 100);
  rect(50, 80, 400, 60, 10); // Rounded corners

  // White text centered in rectangle
  fill(255);
  textSize(22);
  text("Processing + Arduino = Power!", 250, 110);

  // Animated circle
  fill(r, 50, 200);
  ellipse(250, 220, 60, 60);
}
```

Шта овај програм демонстрира?

Концепт	Део програма
Подешавање величине прозора	size(500, 300) in setup()
Континуирана анимација	draw() runs ~60 times per second
Динамичка промена боје помоћу математичких функција	sin() function on frameCount
Цртање правоугаоника	rect(x, y, width, height, radius)
Приказ текста	text(), textSize(), textAlign()
Цртање круга	ellipse(cx, cy, width, height)

✓ За покретање програма притисните Ctrl+R (Windows/Linux) или Cmd+R (Mac). За заустављање програма притисните Ctrl

Вежба 2: Примање података са Arduino

У овој вежби успоставља се комуникација између Processing окружења и Arduino плоче. Arduino шаље бројчану вредност путем серијске комуникације на сваких 500 ms, а Processing примљене податке приказује на екрану.

Корак 1 — Arduino код

```
int counter = 0;
void setup() {
  Serial.begin(9600);
}
void loop() {
  Serial.println(counter); // Send number to Processing
  counter++;
  if (counter > 100) counter = 0;
  delay(500);
}
```

△ Учитајте програм на Arduino и затворите Serial Monitor пре покретања Processing програма.

Корак 2 — Processing код

```
import processing.serial.*;
Serial myPort;
String incomingData = "";
int receivedValue = 0;
void setup() {
  size(400, 250);
  printArray(Serial.list()); // Check available ports
  myPort = new Serial(this, Serial.list()[0], 9600);
}
void draw() {
  background(240);

  // Display received value as text
  fill(50);
  textSize(20);
  text("Received from Arduino: " + receivedValue, 50, 80);

  // Display as a bar
  fill(100, 150, 255);
  float barW = map(receivedValue, 0, 100, 0, 300);
  rect(50, 120, barW, 40);

  // Label on bar
  fill(0);
  textSize(16);
  text(receivedValue + " / 100", 50 + barW/2, 140);
}
void serialEvent(Serial p) {
  incomingData = p.readStringUntil('\n');
  if (incomingData != null) {
    incomingData = trim(incomingData);
    if (incomingData.matches("\\d+")) {
      receivedValue = int(incomingData);
    }
  }
}
```

Објашњење програма

Део програма	објашњење
<code>import processing.serial.*</code>	Увоз библиотеке за серијску комуникацију
<code>Serial myPort</code>	Објекат који представља везу са серијским портом
<code>Serial.list()[0]</code>	Бира први доступан COM порт
<code>serialEvent(Serial p)</code>	Аутоматски се позива када пристигну нови подаци са Arduino плоче
<code>readStringUntil('\n')</code>	Чита један ред текста (до знака за нови ред)
<code>trim()</code>	Уклања размаке и знакове новог реда из текста
<code>matches("\\d+")</code>	Проверава да ли текст садржи само цифре (безбедна конверзија у број)
<code>map(value, 0,100, 0,300)</code>	Пресликава вредност из опсега 0–100 у ширину од 0–300 пиксела

Вежба 3: Обрада више вредности (CSV формат)

У пројектима 1, 3 и 5 Arduino шаље више вредности у једном реду, раздвојених зарезима. Ова вежба показује како се такав формат података обрађује у Processing окружењу.

Arduino код — слање две вредности

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  float temperature = 23.5;    // In real project: sensor reading
  float humidity     = 61.0;
  Serial.print(temperature);
  Serial.print(",");          // Comma separator
  Serial.println(humidity);   // Newline at end
  delay(1000);
}
```

Processing код — обрада две примљене вредности

```
import processing.serial.*;
Serial myPort;
float temperature = 0;
float humidity    = 0;
void setup() {
  size(500, 300);
  myPort = new Serial(this, Serial.list()[0], 9600);
}
void draw() {
  background(255);
  fill(0);
  textSize(22);
  text("Temperature: " + nf(temperature, 1, 1) + " \u00B0C", 50,
100);
  text("Humidity:      " + nf(humidity, 1, 1) + " %", 50, 150);
  // Temperature bar (red)
  fill(220, 50, 50);
  rect(50, 170, map(temperature, 0, 50, 0, 380), 25);
  // Humidity bar (blue)
  fill(50, 100, 220);
  rect(50, 210, map(humidity, 0, 100, 0, 380), 25);
}
void serialEvent(Serial p) {
  String data = p.readStringUntil('\n');
  if (data != null && data.contains(",")) {
    String[] values = split(trim(data), ",");
    if (values.length == 2) {
      temperature = float(values[0]);
      humidity    = float(values[1]);
    }
  }
}
```

Како се обрађују CSV подаци

Корак	Код	Објашњење
1. Читање реда	<code>readStringUntil('\n')</code>	Чека комплетан ред који се завршава знаком за нови ред
2. Провера да вредност није празна	<code>if (data != null)</code>	Спречава грешке приликом читања празних података
3. Провера формата	<code>data.contains(",")</code>	Проверава да ли су подаци у CSV формату
4. Раздвајање текста	<code>split(trim(data), ",")</code>	Раздваја текст у низ елемената на месту зареза
5. Check length	<code>values.length == 2</code>	Ensure correct number of values
6. Конверзија у број	<code>float(values[0])</code>	Претвара текстуалну вредност у број

💡 Овај образац се користи у пројектима 1, 3 и 5. Када га савладате овде, лако ћете га препознати и применити у свим наредним пројектима

Вежба 4: Интеракција са Processingом — слање команди Arduino

Processing може и да шаље податке Arduino плочи. На пример, на овај начин је могуће укључивати и искључивати пумпу (Пројекат 5) или управљати сервомотором (Пројекат 3).

Ова вежба показује како се помоћу кликова мишем и притисака на тастере могу слати команде ка Arduino плочи..

Arduino код — примање команди

```
void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT);          // Built-in LED
}
void loop() {
  if (Serial.available() > 0) {
    char cmd = Serial.read();
    if (cmd == '1') digitalWrite(13, HIGH); // LED ON
    else if (cmd == '0') digitalWrite(13, LOW); // LED OFF
  }
}
```

Processing код — Дугмад миша и тастери као контроле

```
import processing.serial.*;
Serial myPort;
boolean ledOn = false;

void setup() {
  size(400, 220);
  myPort = new Serial(this, Serial.list()[0], 9600);
  textAlign(CENTER, CENTER);
}

void draw() {
  background(240);
  fill(30);
  textSize(18);
  text("Arduino LED Control", 200, 30);
  text("Click buttons or press 1 / 0 on keyboard", 200, 60);

  // ON button
  fill(ledOn ? color(0, 200, 0) : color(180));
  rect(60, 100, 120, 50, 8);
  fill(255);
  text("ON", 120, 125);

  // OFF button
  fill(!ledOn ? color(200, 50, 50) : color(180));
  rect(220, 100, 120, 50, 8);
  fill(255);
  text("OFF", 280, 125);

  // Status
  fill(30);
  textSize(16);
  text("Status: LED is " + (ledOn ? "ON" : "OFF"), 200, 190);
}

void mousePressed() {
  if (mouseX>60&&mouseX<180&&mouseY>100&&mouseY<150)
  { myPort.write('1'); ledOn=true; }
  if (mouseX>220&&mouseX<340&&mouseY>100&&mouseY<150)
  { myPort.write('0'); ledOn=false; }
}

void keyPressed() {
  if (key=='1') { myPort.write('1'); ledOn=true; }
  if (key=='0') { myPort.write('0'); ledOn=false; }
}
```

Задаци за ученике

Ниво	Задатак	Вештине које се увежбавају
● основни	Приказати једну бројчану вредност са Arduino плоче као текст	Serial, parseInt, text()
● средњи	Нацртати обојени стуб чија је дужина пропорционална вредности са сензора	map(), rect(), fill()
● напредни	Обрадити две вредности (CSV формат) и приказати обе помоћу стубичастих приказа	split(), multiple visual elements
● изазов	Додати ON/OFF дугмад која шаљу команде Arduino плочи	mousePressed(), Serial.write()
● изазов	Направити графикон у реалном времену за последњих 100 мерења	Arrays, shifting values, line()

Честе грешке и решавање проблема

Грешка / Проблем	Највероватнији узрок	Решење
Порт није пронађен / нема Serial.list()	Arduino није повезан или недостаје драјвер	Проверити USB везу и инсталирати CH340 драјвер ако је потребно
Порт је заузет	Arduino Serial Monitor је и даље отворен	Затворити Serial Monitor у Arduino IDE
Нема примљених података	Погрешан индекс у Serial.list()[0]	Покренути printArray(Serial.list()) и проверити индекс
Неусклађена брзина комуникације	Arduino користи 9600, а Processing другу вредност	Подесити Serial.begin(9600) и new Serial(...,9600)
NullPointerException	Променљива data је празна (непотпун ред)	Увек проверити: if (data != null)
Неисправне вредности на екрану	Неуспешна конверзија текста у број	Додати проверу: if (data.matches("\\d+"))
Processing програм престаје да реагује	Функција draw() је преоптерећена сложеним прорачунима	Читање серијских података преместити у serialEvent()

✓ Златно правило: Ако ништа не функционише, поново покрените и Arduino и Processing. Већина проблема са серијском комуникацијом решава се поновним успостављањем везе

Processing и Arduino IDE — поређење

Концепт	Arduino IDE	Processing
Главне функције програма	setup() + loop()	setup() + draw()
Испис у конзолу	Serial.println()	println()
Читање улаза	Serial.read()	myPort.readStringUntil()
Пауза у извршавању	delay(ms)	delay(ms) (avoid in draw())
Типови променљивих	int, float, char, bool	int, float, String, boolean
Цртање правоугаоника	нема графички приказ	rect(x, y, w, h)
Испис текста	нема графички приказ	text(str, x, y)
Рад са бојама	нема графички приказ	fill(r, g, b) / stroke(r, g, b)
Обрада догађаја	—	mousePressed(), keyPressed()
Увоз библиотека	#include	import library.*

Практична примена

Визуелизација података помоћу Processing окружења користи се у:

- индустријским контролним таблама и системима за праћење података у реалном времену;
- научној визуелизацији података у истраживачким лабораторијама;
- интерактивним музејским поставкама и образовним инсталацијама;
- креативном програмирању и генеративној дигиталној уметности;
- новинарству заснованом на подацима и визуелизацији информација;
- системима за праћење здравствених параметара и biofeedback уређајима;
- системима за праћење стања животне средине (квалитет ваздуха, метеоролошке станице);
- развоју прототипова игара и интерактивних мултимедијалних садржаја.

Међупредметна повезаност

Предмет	Повезаност	Пример у пројектима
Информатика	Програмирање, структуре података, обрада догађаја	serialEvent(), низови за графиконе
Математика	Координатни систем, скалирање, функције	map(), sin(), приказ података на графикону
Физика	Интерпретација података са сензора, мерења	Приказ температуре, радарски приказ растојања
Ликовна култура и дизајн	Теорија боја, композиција, дизајн	Дизајн корисничког интерфејса, прикази у боји
Техника и технологија	Системски приступ, интеграција хардвера и софтвера	Комуникација између Arduino и Processing окружења

DigComp компетенције

Ова наставна јединица развија следеће компетенције према оквиру DigComp 2.2:

DigComp област	Развијена компетенција
1. Информациона и податковна писменост	Пријем, обрада и приказ података са сензора у реалном времену
2. Комуникација и сарадња	Осмишљавање јасних визуелних интерфејса за представљање података
3. Креирање дигиталних садржаја	Писање Processing програма и израда визуелних апликација
4. Безбедност	Разумевање провере података и обраде грешака
5. Решавање проблема	Отклањање проблема у серијској комуникацији и логици визуелизације

Посебно се развијају:

- рачунарско размишљање кроз програмирање засновано на догађајима;
- податковна писменост – претварање бројчаних података у визуелно значење;
- вештине креативног програмирања и дизајна корисничког интерфејса;
- систематично решавање проблема и отклањање грешака;
- системско размишљање кроз повезивање хардвера и софтвера.

Идеје за проширење пројекта

Ученици који савладају основне концепте могу проширити своја знања у Processing окружењу на следеће начине:

- цртањем графикона у реалном времену за последњих 100 мерења;
- додавањем звучних упозорења помоћу библиотеке Minim;
- чувањем података са сензора у CSV датотеку коришћењем класе PrintWriter;
- израдом контролне табле са приказом више сензора;
- додавањем часовника и временске ознаке сваком забележеном мерењу;
- израдом web верзије помоћу p5.js (JavaScript верзија Processing окружења);
- повезивањем више Arduino плоча са једним Processing програмом.

Прошињере	Ниво сложености	Кључни концепт
Графикон у реалном времену	● Средњи	Низови, померање вредности, line()
Чување података у CSV фајл	● Средњи	PrintWriter, saveStrings()
Звучни аларм при прекорачењу прага	● Напредни	Библиотека Minim, репродукција звука

Прошињере	Ниво сложености	Кључни концепт
Контролна табла за више сензора	● Напредни	Више серијских портова или мултиплексирани подаци
Web верзија у p5.js окружењу	● Изазов	JavaScript, HTML Canvas, WebSerial API

Каква је веза између овог увода и наредних пројеката?

Пројекат	Сензор	Визуелизација у Processing окружењу	Основне технике
1 DHT11	Температура и влажност	Приказ две вредности текстом и стубичастим приказима	CSV обрада података, две вредности
2 LDR	Интензитет осветљености	Један стубичасти приказ и промена боје	map(), background color
3 HC-SR04	Растојање, сонар	Радарски приказ	Тригонометрија, ellipse, line
4 Мерач отпорности	Отпорност	Бројчани приказ и мерач	nf(), arc(), text
5 Аутоматско наводњавање	Влажност тла	Приказ статуса и стубичасти приказ	Логика прагова, боја испуне

✓ Свих пет пројеката користе исти основни образац рада: `serialEvent()` → обрада података → `draw()`. Када савладате овај образац у овом уводном делу, моћи ћете да га примените у свим наредним пројектима.

Наредни кораци

Након завршетка овог уводног поглавља, препоручује се наставак рада следећим редоследом:

Корак	Пројекат	Нова знања и вештине у Processing окружењу
Увод	Увод у Arduino	Основе серијске комуникације, <code>setup()</code> и <code>loop()</code>
← Налазите се овде	Увод у Processing	<code>setup()</code> , <code>draw()</code> , <code>serialEvent()</code> , <code>map()</code> , графички облици
Пројекат 1	DHT11 – температура и влажност ваздуха	CSV обрада података, приказ две вредности
Пројекат 2	Сензор осветљености – LDR	Приказ једне вредности, динамичка промена боје позадине
Пројекат 3	Сонар – HC-SR04	Радарски приказ, тригонометрија, управљање сервомотором
Пројекат 4	Мерач отпорности	Форматирање бројева, визуелизација

Корак	Пројекат	Нова знања и вештине у Processing окружењу
		помоћу мерача
Пројекат 5	Аутоматско наводњавање	Приказ статуса система, кориснички интерфејс за управљање пумпом

Закључак

Processing представља визуелну везу између физичког света и екрана рачунара. Када ученици разумеју како се подаци преносе од сензора, преко серијске комуникације Arduino плоче, до функције `draw()` у Processing окружењу, стичу основу за израду било које визуелизације података у реалном времену.

Овај увод обухвата све што је потребно за реализацију свих пет пројеката у оквиру радионица „Дигитални изазов“: успостављање серијске комуникације, пријем и обраду података, приказ текста и графичких облика, употребу функције `map()` за скалирање вредности и слање команди ка Arduino плочи.

Ове вештине, у комбинацији са знањима стеченим у уводном поглављу о Arduino платформи, представљају комплетан STEM алат за реализацију интерактивних пројеката заснованих на сензорима.

Наредни кораци

Након завршетка овог уводног поглавља, препоручује се наставак рада следећим редоследом

Корак	Пројекат	Нова знања и вештине у Processing окружењу
Увод	Увод у Arduino	Основе серијске комуникације, <code>setup()</code> и <code>loop()</code>
← Налазите се овде	Увод у Processing	<code>setup()</code> , <code>draw()</code> , <code>serialEvent()</code> , <code>map()</code> , графички облици
Пројекат 1	DHT11 Температура и влажност	CSV обрада података, приказ две вредности
Пројекат 2	Сензор осветљености — LDR	Приказ једне вредности, динамичка промена боје позадине
Пројекат 3	Сонар — HC-SR04	Радарски приказ, тригонометрија, управљање сервомотором
Пројекат 4	Мерач отпорности	Форматирање бројева, визуелизација помоћу мерача
Пројекат 5	Аутоматско наводњавање	Приказ статуса система, кориснички интерфејс за управљање пумпом

Пројекат 1: Температура и влажност — DHT11

Опис

У овој вежби ученици праве једноставан систем за мерење температуре и влажности ваздуха помоћу Arduino Uno микроконтролера и DHT11 сензора. Прикупљени подаци се путем серијске комуникације преносе на рачунар и приказују у реалном времену коришћењем Processing окружења.

Вежба повезује основе електронике, програмирања, физике и визуелизације података и представља увод у IoT (Internet of Things) системе и рад са сензорима.

Циљеви

Ученици ће:

- научити како функционишу дигитални сензори;
- повезати DHT11 сензор са Arduino микроконтролером;
- читавати податке са сензора;
- користити серијску комуникацију;
- визуелизовати податке у реалном времену;
- развити једноставан систем упозорења заснован на унапред дефинисаним граничним вредностима..

Резултати

Након радионице ученици ће бити у стању да:

- правилно повежу дигитални сензор са микроконтролером;
- користе библиотеке у окружењу Arduino IDE;
- читавају и обрађују податке са сензора;
- разумеју принцип серијске комуникације;
- визуелизују податке у Processing окружењу;
- анализирају промене температуре и влажности ваздуха;
- самостално прошире систем додавањем нових функционалности.

Потребна опрема

Компонента	Количина	Улога
Arduino Uno	1	Микроконтролер
DHT11 сензор	1	Сензор за мерење температуре и влажности
10 kΩ отпорник	1	Стабилност комуникације, pull up отпорник
Breadboard плоча	1	Повезивање компоненти
Проводници	~5	Повезивање компоненти
USB кабл	1	Веза са компјутером
Arduino IDE	1	Програмирање Arduina
Processing	1	Визуелизација података

Теоријска основа

Шта је DHT11?

DHT11 је дигитални сензор који мери температуру и релативну влажност ваздуха и шаље податке у дигиталном облику. Погодан је за почетнике јер не захтева аналогно-дигиталну конверзију, једноставно се повезује и за њега су доступне готове програмске библиотеке.

Шта је релативна влажност ваздуха?

Релативна влажност представља процентуални однос количине водене паре у ваздуху и максималне количине водене паре коју ваздух може да садржи при датој температури и нормалном атмосферском притиску.

Серијска комуникација

Arduino шаље податке рачунару путем серијске комуникације преко USB порта. На пример:

```
Serial.println("24.5,60");
```

То значи да је: температура = 24.5 °C, влажност = 60 %

Вежба 1: Повезивање са DHT11 сензором

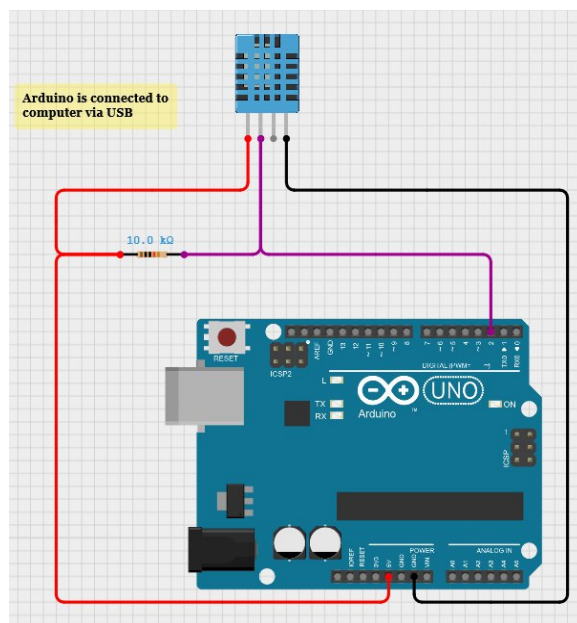
Wiring

DHT11 Pin	Arduino Pin
VCC	5V
GND	GND
DATA	Дигитални пин 2

Уколико користите самостални („raw“) DHT11 сензор:

- поставите отпорник од 10 kΩ између пинова VCC и DATA.

✓ **Савет:** DHT11 модул са 4 пина већ има уграђен pull-up отпорник.



Arduino програм

Корак 1: Инсталирање библиотека

У Arduino IDE окружењу:

- Sketch → Include Library → Manage Libraries
- Потражити: "DHT sensor library" од Adafruit-a
- Инсталирати библиотеку

Корак 2: Arduino код

```
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  Serial.begin(9600);
  dht.begin();
}
void loop() {
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature(); // Celsius

  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  Serial.print(temperature);
  Serial.print(",");
  Serial.println(humidity);

  delay(2000);
}
```

Објашњење кода

Команда	Објашњење
dht.readTemperature()	Чита температуру
dht.readHumidity()	Чита влажност
Serial.print()	Шаље податке компјутеру
delay(2000)	Чека две секунде

✓ Подаци се шаљу у облику вредности раздвојених зарезима (CSV формат), јер их Processing једноставно прима и обрађује.

Вежба 2: Визуелизација података у Processing окружењу

Processing код

Шта приказује програм?

```
Serial myPort;
String data;
float temp, hum;

void setup() {
  size(400, 200);
  myPort = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
  background(255);
  fill(0);
  textSize(20);
  text("Temperature: " + temp + " °C", 50, 80);
  text("Humidity: " + hum + " %", 50, 120);

  if (myPort.available() > 0) {
    data = myPort.readStringUntil('&apos;\n&apos;');
    if (data != null && data.contains(",")) {
      String[] values = split(trim(data), ",");
      if (values.length == 2) {
        temp = float(values[0]);
        hum = float(values[1]);
      }
    }
  }
}
```

- Чита податке које шаље Arduino,
- Приказује температуру и влажност,

Задаци за ученике

Level	Task
● Основни	Прикажи температуру и влажност
● Средњи	Додај траке напретка (progress bar)
● Напредни	Додај аларм за високу температуру
● Изазов	Сачувај податке у CSV фајл
● Изазов	Прикажи на графикону у реалном времену вредности

Пример система за упозоравање

```
if (temperature > 30) {
  fill(255, 0, 0);
  text("WARNING: HIGH TEMPERATURE!", 40, 50);
}
```

Решавање проблема

Проблем	Могући узрок
Нема примљених података	Неодговарајући/неисправан COM порт
"Sensor error" порука	Пинови сензора су погрешно повезани
Нестабилне вредности са сензора	Лош контакт на breadboard плочи или кабловима
Processing не прима податке	Није усклађена брзина серијске комуникације

✓ Савет: Искористи `Serial.list()` да пронађеш одговарајући COM порт.

Примене у пракси

Овакви системи користе се у:

- паметним кућама;
- метеоролошким станицама;
- пластеницима и стакленицима;
- серверским просторијама;
- IoT системима;
- индустријској аутоматизацији.

Међупредметна повезаност

Предмет	Веза
Физика	Мерење, температура, релативна влажност
Информатика	Програмирање и обрада података
Техника и технологија	Електроника и сензори
Математика	Анализа података и графички приказ резултата

DigComp компетенције

Ова радионица развија:

- дигиталну писменост;
- вештине решавања проблема;
- анализу података;
- програмерске вештине;
- сарадњу у STEM окружењу.

Идеје за проширење пројекта

- додавање OLED дисплеја;
- слање података путем WiFi мреже;
- израда web контролне табле (dashboard);
- повезивање система са cloud платформом;
- израда мини метеоролошке станице.

Закључак

Ова вежба представља увод у рад са сензорима, микроконтролерима и визуелизацијом података. Комбиновањем Arduino Uno платформе и Processing окружења ученици стичу практична знања из електронике, програмирања и анализе података кроз реалан STEM пројекат.

Пројекат 2: Сензор осветљености — Фотоотпорник (LDR)

Опис

У овом пројекту мери се интензитет амбијенталног осветљења помоћу фото отпорника (LDR) повезаног са Arduino плочом. Аналогна вредност (0–1023) шаље се путем серијске комуникације у Processing окружење, где се у реалном времену приказује као бројчана вредност и графичка трака.

Циљеви

Ученици ће:

- разумети како фотоотпорник (LDR) региструје промене интензитета светлости;
- научити како аналогни сензори функционишу са Arduino Uno микроконтролером;
- направити коло са напонским делиоцем;
- читавати и тумачити вредности аналогног сензора;
- користити серијску комуникацију између Arduino плоче и рачунара;
- визуелизовати податке са сензора у реалном времену помоћу Processing окружења;
- развити једноставне интерактивне визуелизације засноване на подацима из окружења.

Резултати учења

Након завршене радионице ученици ће бити у стању да:

- правилно повежу LDR сензор помоћу кола са напонским делиоцем;
- читавају аналогне вредности са аналогног улаза Arduino плоче;
- објасне везу између интензитета светлости и електричне отпорности;
- преносе податке са сензора путем серијске комуникације;
- израђују једноставне визуелизације података у реалном времену у Processing окружењу;
- анализирају утицај услова осветљења на читавања сензора;
- проширују пројекат додавањем нових визуелних или интерактивних функционалности.

Потребна опрема

Компонента	Количина	Намена
Arduino Uno	1	Микроконтролер
Фотоотпорник (LDR)	1	Сензор осветљености
10 kΩ отпорник	1	Разделник напона
Breadboard плоча	1	Повезивање компоненти
Проводници	~5	Повезивање компоненти
USB Кабл	1	Повезивање са компјутером
Processing	1	Визуелизација података

Теоријске основе

Како функционише фотоотпорник?

Фотоотпорник (LDR) мења своју електричну отпорност у зависности од количине светлости која пада на његову површину. При јачем осветљењу отпорност се смањује, док се у мраку повећава.

Разделник напона

LDR је повезан редно са фиксним отпорником од 10 kΩ. Напон у тачки између ова два отпорника, који се доводи на аналогни улаз A0, мења се у зависности од интензитета светлости.

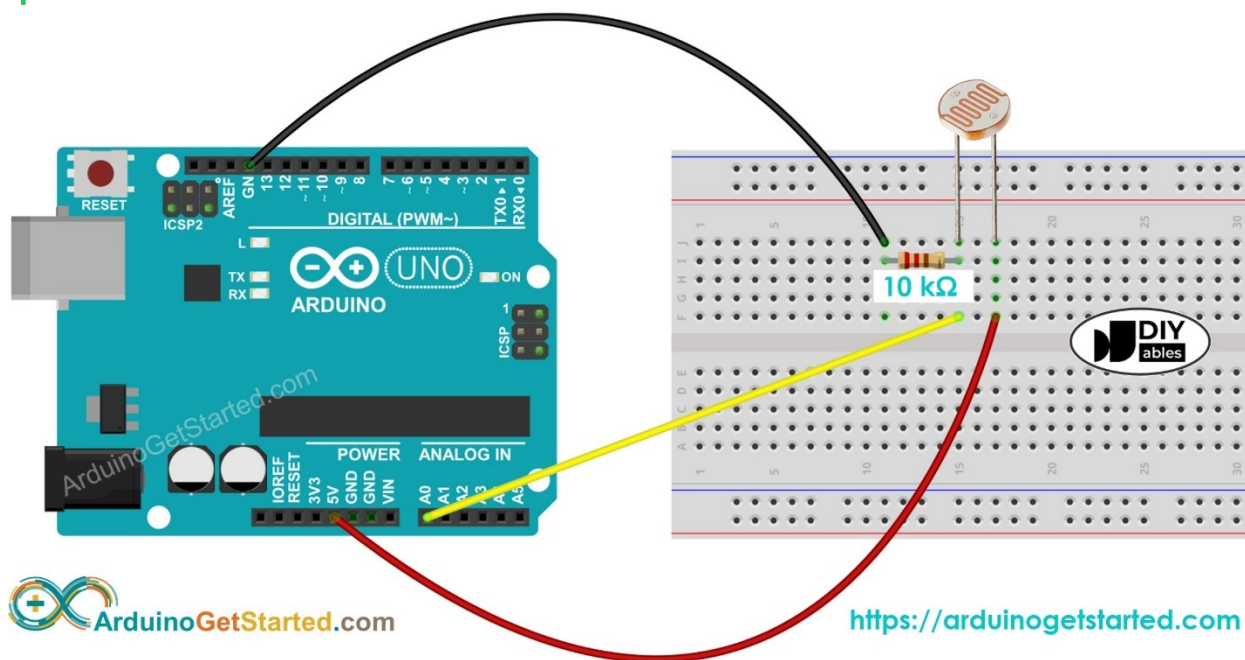
💡 $V_{out} = V_{in} \times R_2 / (R_1 + R_2)$ где је R2 фотоотпорник (LDR).

Вежба 1: Повезивање кола и Arduino програм

Повезивање

Компонента	Повезује се са
Фотоотпорник, LDR (ножица 1)	5V
Фотоотпорник LDR (ножица 2)	A0 и 10 kΩ отпорник (ножица 1)
10 kΩ отпорник (ножица 2)	GND

✓ Спој фотоотпорника и отпорника је повезан са пином A0 на Arduino.



Arduino код

```
const int sensorPin = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int lightValue = analogRead(sensorPin); // 0 (dark) to 1023 (bright)
  Serial.println(lightValue);
  delay(200);
}
```

Вежба 2: Визуелизација у Processing окружењу

Processing код — испис текста

```
import processing.serial.*;
Serial myPort;
String val = "";
int lightLevel = 0;

void setup() {
  size(400, 200);
  myPort = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
  background(255);
  fill(0);
  textSize(20);
  text("Light Intensity: " + lightLevel, 50, 100);

  if (myPort.available() > 0) {
    val = myPort.readStringUntil('\n');
    if (val != null) { lightLevel = int(trim(val)); }
  }
}
```

Processing код — Bar + Serial Event

```
import processing.serial.*;
Serial myPort;
String val = "";
int lightLevel = 0;

void setup() {
  size(400, 200);
  myPort = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
  background(255);
  fill(0);
  textSize(20);
  text("Light Intensity: " + lightLevel, 50, 50);

  fill(0, 100, 255);
  rect(50, 100, map(lightLevel, 0, 1023, 0, 300), 30);
}

void serialEvent(Serial p) {
  val = p.readStringUntil('\n');
  if (val != null) {
    val = trim(val);
    if (val.matches("\\d+")) { lightLevel = int(val); }
  }
}
```

Задаци за ученике

Ниво	Задатак
● Основни	Приказати бројчану вредност интензитета осветљености
● Средњи	Нацртати обојену траку чија је дужина пропорционална интензитету осветљености
● Напредни	Мењати боју позадине у зависности од интензитета осветљености
● Изазов	Чувати измерене вредности са временском ознаком у текстуалном фајлу

Решавање проблема

Проблем	Могући узрок
Вредност је стално 0	Фотоотпорник није повезан или је коло у прекиду
Вредност је стално 1023	Отпорник није исправно повезан
Нестабилна читавања	Лош контакт на breadboard плочи

Идеје за проширење пројекта

- додавање LED диоде која аутоматски реагује на интензитет осветљености;
- израда аутоматског ноћног светла;
- употреба више LDR сензора за одређивање правца из којег долази светлост;
- повезивање система са WiFi модулом ради удаљеног праћења;
- израда једноставног прототипа паметног осветљења;
- приказ података на OLED или LCD дисплеју;
- бележење података о осветљености ради дугорочне анализе.

Закључак

Овај пројекат упознаје ученике са аналогним сензорима, колима са напонским делиоцем и визуелизацијом података у реалном времену. Комбиновањем Arduino Uno платформе и Processing окружења ученици стичу практична искуства из електронике, програмирања и интерактивних система заснованих на сензорима кроз практичну STEM активност.

Project 3: Sonar — Ultrasonic Sensor HC-SR04

Опис

У овом пројекту користи се ултразвучни сензор HC-SR04 за мерење растојања. Подаци се визуелизују у Processing окружењу кроз три постепено сложеније вежбе: текстуални приказ, управљање сервомотором и комплетан радарски/сонарски приказ.

Циљеви

Ученици ће:

- разумети принцип рада ултразвучног мерења растојања;
- научити како HC-SR04 ултразвучни сензор мери растојање помоћу звучних таласа;
- повезати и програмирати ултразвучне сензоре и сервомоторе помоћу Arduino Uno плоче;
- користити серијску комуникацију за пренос измерених података;
- визуелизовати податке са сензора у реалном времену помоћу Processing окружења;
- развијати интерактивне визуелизације у облику радара и сонара;
- истраживати повезаност физике, електронике и програмирања кроз практичан STEM пројекат.

Резултати учења

Након завршене радионице ученици ће бити у стању да:

- објасне принцип рада ултразвучних сензора;
- израчунају растојање на основу времена повратка еха;
- правилно повежу и програмирају HC-SR04 сензор;
- управљају сервомотором помоћу Arduino плоче;
- преносе податке са сензора у Processing окружење путем серијске комуникације;
- креирају графичке приказе мерења растојања у реалном времену;
- развијају једноставне интерфејсе налик радару;
- анализирају утицај положаја сензора и удаљености објекта на резултате мерења;

самостално проширују систем додавањем нових функција и интеракција

Теоријске основе

Физичке основе ултразвучног мерења

Ултразвучни сензори емитују звучне таласе високе фреквенције (око 40 kHz). Када таласи наиђу на препреку, одбијају се и враћају назад до сензора. Сензор мери време потребно да се одбијени талас врати. Растојање се израчунава по формули:

:

$$\text{Distance} = (\text{Speed of Sound} \times \text{Time}) / 2$$

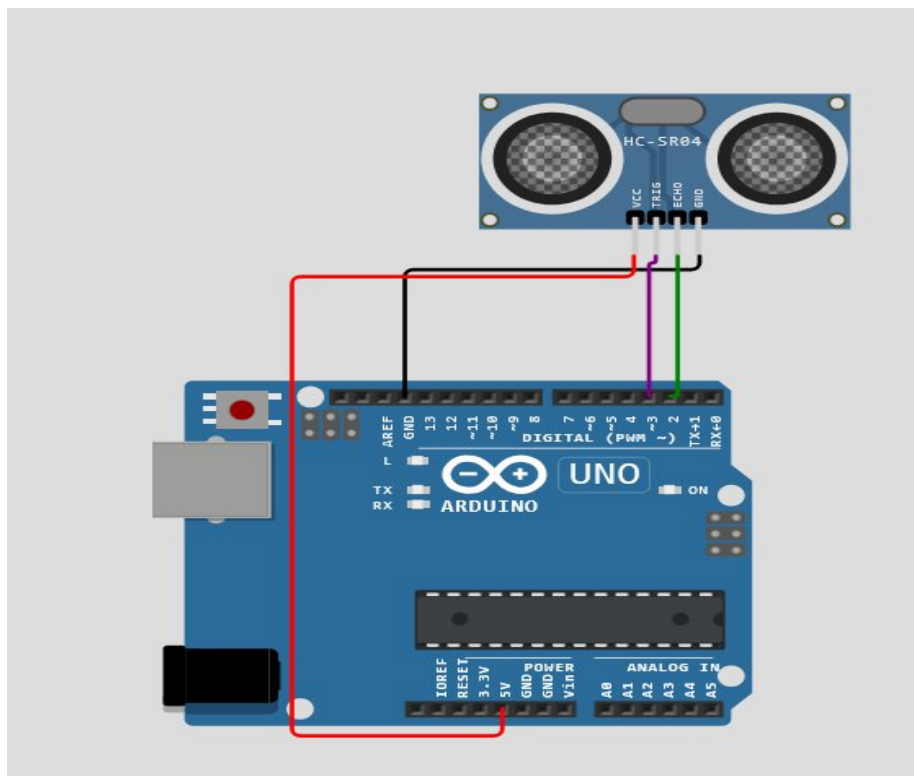
Брзина звука у ваздуху износи приближно 343 m/s. Дељење са 2 узима у обзир пут који талас прелази до објекта и назад.

Потребна опрема

Компонента	Количина	Улога
Arduino Uno	1	Микроконтролер
HC-SR04 ултразвучни сензор	1	Мерење растојања
SG90 серво мотор	1	Окретање „радара“ (Вежбе. 2 и 3)
Пластични држач сензора	1	Монтажа сензора на серво мотор
Breadboard + жице	1 комплет	Повезивање компоненти
USB кабл	1	Connection to computer
Processing	1	Визуелизација

Вежба 1: Мерење растојања и приказ у Processing-у

Повезивање



VCC 5V
 GND GND
 TRIG Pin 3
 ECHO Pin 2

Arduino код

```
#define trigPin 3
#define echoPin 2
void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}
void loop() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH);
  float distance = duration * 0.0343 / 2;
  Serial.println(distance);
  delay(200);
}
```

Processing код – приказ растојања путем текста

```
import processing.serial.*;
Serial myPort;
String val;
void setup() {
  size(400, 200);
  myPort = new Serial(this, Serial.list()[0], 9600);
}
void draw() {
  if (myPort.available() > 0)
  {
    val = myPort.readStringUntil('\n');
    if (val != null) {
      background(255);
      fill(0);
      textSize(20);
      text("Distance: " + val.trim() + " cm", 50, 100);
    }
  }
}
```

Processing код – приказ растојања путем хоризонталног бара

```
import processing.serial.*;
Serial myPort;
float distance;

void setup() {
  size(600, 200);
  printArray(Serial.list());
  myPort = new Serial(this, Serial.list()[0], 9600);
  myPort.bufferUntil('\n');
}

void draw() {
  background(255);
  fill(0);
  textSize(24);
  text("Distance: " + nf(distance, 0, 2) + " cm", 50, 50);
  // Horizontal bar
  fill(100, 200, 255);
  rect(50, 100, map(distance, 0, 200, 0, width - 100), 30);
}

void serialEvent(Serial myPort) {
  String data = myPort.readStringUntil('\n');
  if (data != null) {
    data = trim(data);
    distance = float(data);
  }
}
```

Вежба 2: Контрола SG90 серво мотора из Processing-a

Компоненте

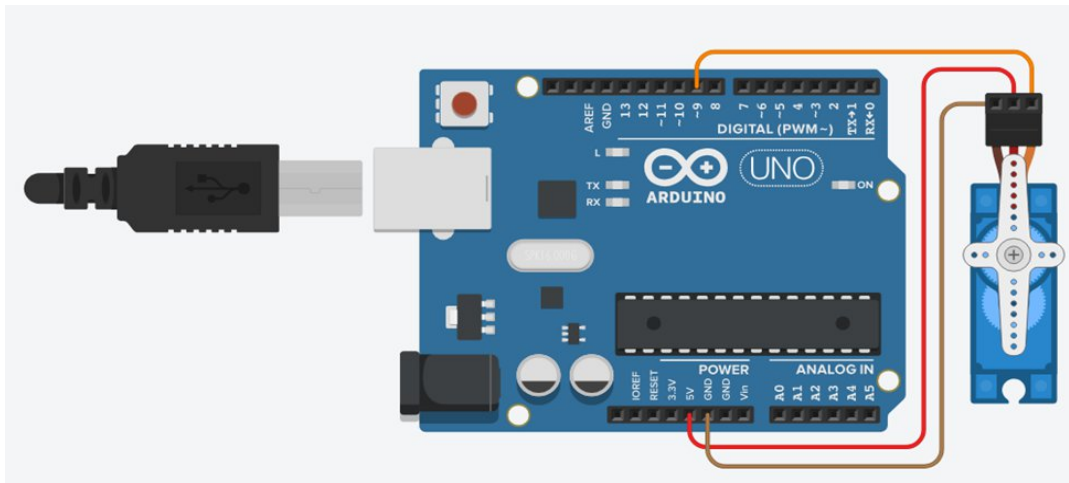
- Arduino Uno
- SG90 Servo motor
- USB кабл

Повезивање

Пин серво мотора	Arduino пин
Сигнал (наранџаст)	Пин 9
VCC (црвен)	5V
GND (браон/црн)	GND

[The Beginners Guide to Micro Servos](https://docs.arduino.cc/learn/electronics/servo-motors/)

<https://docs.arduino.cc/learn/electronics/servo-motors/>



Arduino код

```
#include <Servo.h>
Servo servo;
String command = "";
bool sweeping = false;
int angle = 0;
void setup() {
  Serial.begin(9600);
  servo.attach(9);
}
void loop() {
  if (Serial.available()) {
    command = Serial.readStringUntil('\n');
    command.trim();
    if (command == "1") sweeping = true;
    else if (command != "1") sweeping = false;
  }
  if (sweeping) {
    for (angle = 0; angle <= 180 && sweeping; angle++) {
      servo.write(angle);
      delay(10);
    }
    for (angle = 180; angle >= 0 && sweeping; angle--) {
      servo.write(angle);
      delay(10);
    }
  }
}
```

Processing Code — START / STOP Buttons

```
import processing.serial.*;

Serial myPort;
boolean isSweeping = false;

void setup() {
  size(400, 200);
  myPort = new Serial(this, Serial.list()[0], 9600);
  textAlign(CENTER, CENTER);
}

void draw() {
  background(240);
  textSize(16);
  fill(0);
  text("Servo Control", width/2, 30);

  drawButton("START", 100, 100, isSweeping);
  drawButton("STOP", 250, 100, !isSweeping);
}

void drawButton(String label, int x, int y, boolean active) {
  fill(active ? color(0, 200, 0) : color(180));
  rect(x, y, 100, 40, 7);
  fill(0);
  text(label, x + 50, y + 20);
}

void mousePressed() {
  if (mouseX > 100 && mouseX < 200 && mouseY > 100 && mouseY < 140) {
    myPort.write("1\n");
    isSweeping = true;
  } else if (mouseX > 250 && mouseX < 350 && mouseY > 100 && mouseY < 140) {
    myPort.write("0\n");
    isSweeping = false;
  }
}
```

Како ради

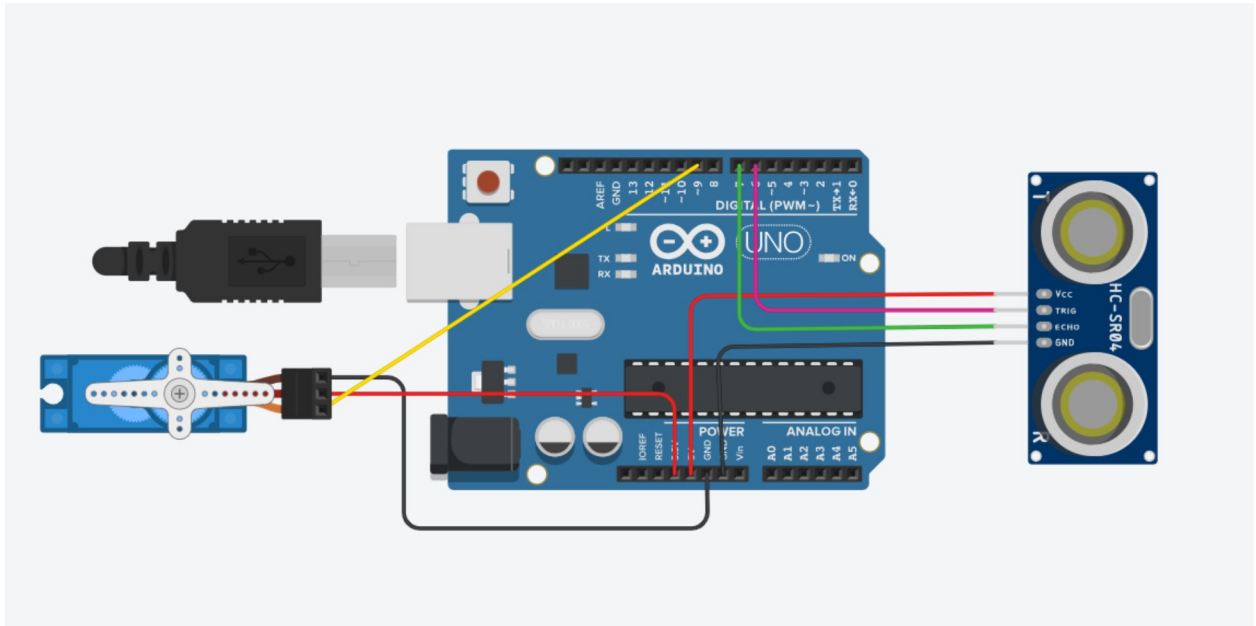
- **Start button** шаље "START" → Arduino покреће серво.
- **Stop button** шаље "STOP" → Arduino зауставља кретање серва.
- Можете проширити овај пример додавањем слајдера за дефинисање угла кретања серво мотора, контролом брзине.

Вежба 3: Ултразвучни сензор контролише серво мотор

Сервомотор се окреће на 90° када се детектује објекат на растојању мањем од 50 cm, а у супротном се враћа у положај од 0° .

Arduino код

```
#include <Servo.h>
// constants won't change
const int TRIG_PIN = 6; // Pin connected to Ultrasonic Sensor's TRIG pin
const int ECHO_PIN = 7; // Pin connected to Ultrasonic Sensor's ECHO pin
const int SERVO_PIN = 9; // Arduino pin connected to Servo Motor's pin
const int DISTANCE_THRESHOLD = 50; // centimeters
Servo servo; // create servo object to control a servo
// variables will change:
float duration_us, distance_cm;
void setup() {
    Serial.begin(9600); // initialize serial port
    pinMode(TRIG_PIN, OUTPUT); // set arduino pin to output mode
    pinMode(ECHO_PIN, INPUT); //set arduino pin to input mode
    servo.attach(SERVO_PIN); // attaches the servo on pin 9 to the servo
    object
    servo.write(0);
}
void loop() {
    // generate 10-microsecond pulse to TRIG pin
    digitalWrite(TRIG_PIN, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);
    // measure duration of pulse from ECHO pin
    duration_us = pulseIn(ECHO_PIN, HIGH);
    // calculate the distance
    distance_cm = 0.017 * duration_us;
    if(distance_cm < DISTANCE_THRESHOLD)
        servo.write(90); // rotate servo motor to 90 degree
    else servo.write(0); // rotate servo motor to 0 degree
    // print the value to Serial Monitor
    Serial.print("distance: ");
    Serial.print(distance_cm);
    Serial.println(" cm");
    delay(500);
}
```



Компоненте

- Arduino Uno
- HC-SR04 ultrasonic sensor
- SG90 servo motor
- Plastic holder (to rotate sensor)
- Processing IDE

Tutorial Reference

[Arduino Ultrasonic Sensor + Servo Motor Tutorial](#)

Задатак за ученике

Напишите Processing програм који приказује измерено растојање у два визуелна формата:

- а) као бројчану вредност помоћу функције `text()`;
- б) као хоризонталну траку пропорционалну растојању.

Вежба 4: Направи ултразвучни радар (сонар)

🔧 Ова вежба је погодна за пројектни задатак.

Arduino код — покретање радара

```
#include <Servo.h>
Servo myServo;
#define trigPin 9
#define echoPin 10
void setup() {
  Serial.begin(9600);
  myServo.attach(6);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  for (int angle = 0; angle <= 180; angle += 2) {
    myServo.write(angle);
    delay(50);

    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    long duration = pulseIn(echoPin, HIGH);
    float distance = duration * 0.0343 / 2;

    Serial.print(angle);
    Serial.print(",");
    Serial.println(distance);
  }
}
```

Processing код — Приказ радара

```
import processing.serial.*;

Serial myPort;
String data;
int angle;
float distance;

void setup() {
  size(600, 600);
  myPort = new Serial(this, Serial.list()[0], 9600);
  background(0);
}

void draw() {
  if (myPort.available() > 0) {
    data = myPort.readStringUntil('&apos;\n&apos;');
    if (data != null && data.contains(",")) {
      String[] parts = split(trim(data), ",");
      if (parts.length == 2) {
        angle = int(parts[0]);
        distance = float(parts[1]);

        float r = map(distance, 0, 100, 0, 250);
        float x = width/2 + r * cos(radians(angle));
        float y = height - r * sin(radians(angle));

        stroke(0, 255, 0);
        fill(0, 255, 0);
        ellipse(x, y, 5, 5);
      }
    }
  }
}
```

Задачи за ученике

Ниво	Задатак
● Основни	Приказ растојања као текста на екрану радара
● Средњи	Цртање помоћних линија и кругова на приказу радара
● Напредни	Обојити тачке према растојању (зелена/жута/црвена)
● Challenge	Додати анимацију линије скенирања и звучно упозорење.

Идеје за проширење пројекта

- додавање звучних ефеката или аларма приликом детекције објекта;
- израда потпуно анимираног радарског интерфејса;
- додавање више ултразвучних сензора ради покривања ширег простора;
- повезивање система са WiFi модулом ради удаљеног праћења;
- израда прототипа паметног помоћника за паркирање;
- интеграција LED диода за приказ растојања до објекта;
- чување радарских података ради касније анализе;
- развој једноставног робота за избегавање препрека;
- комбиновање система са температурним или светлосним сензорима ради вишенаменског надзора.

Закључак

Овај пројекат упознаје ученике са ултразвучним сензорима, системима за мерење у реалном времену, управљањем сервомоторима и интерактивном визуелизацијом података. Комбиновањем HC-SR04 ултразвучног сензора, Arduino Uno платформе и Processing окружења ученици стичу практична искуства из физике, електронике, програмирања и пројектовања интерактивних система кроз практичан STEM пројекат инспирисан стварним радарским и сонарским технологијама.

Пројекат 4: Arduino мерач отпорности са приказом вредности у Processing-у

Опис

У овом пројекту користи се коло са напонским делиоцем за мерење непознатог отпорника. Arduino израчунава вредност отпорности применом Омовог закона и резултат шаље у Processing окружење, где се он приказује у реалном времену.

Циљеви

Ученици ће:

- разумети принцип рада напонског делиоца и примену Омовог закона;
- научити како да измере непознату отпорност помоћу Arduino Uno плоче;
- користити аналогне улазе за мерење напона;
- израчунавати вредност отпорности на основу измереног напона;
- преносити израчунате податке путем серијске комуникације;
- визуелизовати резултате мерења у реалном времену помоћу Processing окружења;
- развити практично разумевање електричних мерних система.

Резултати учења

Након завршене радионице ученици ће бити у стању да:

- објасне принцип рада кола са напонским делиоцем;
- примене Омов закон за израчунавање непознате отпорности;
- правилно повежу отпорнике у конфигурацији напонског делиоца;
- користе аналогне улазе Arduino плоче за мерење напона;
- обрађују и преносе мерне податке путем серијске комуникације;
- креирају једноставне визуелизације у реалном времену у Processing окружењу;
- тумаче утицај вредности отпорника на напон у колу;
- самостално унапређују пројекат додавањем нових могућности за мерење или визуелизацију.

Потребна опрема

Component	Quantity	Purpose
Arduino Uno	1	Microcontroller
Отпорник познате отпорности (1kΩ)	1	Референтни отпорник разделник напона
Отпорник непознате отпорности	1	Мери се вредност отпорности
Breadboard + жице	1 комплет	Повезивање компоненти
USB кабл	1	Веза са компјутером
Processing IDE	1	Приказ

Теоријске основе

Разделник напона и Омов закон

Напонски делилац се састоји од два отпорника повезана редно. Напон у тачки између отпорника зависи од односа њихових отпорности.

:

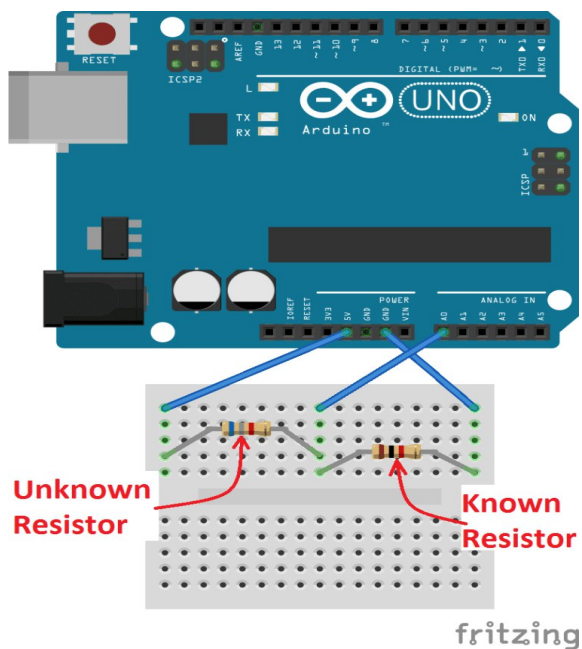
$$V_{out} = V_{in} \times R_2 / (R_1 + R_2) \rightarrow R_2 = R_1 \times (V_{in}/V_{out} - 1)$$

Arduino читава напон V_{out} преко аналогног улаза A0 и на основу измерене вредности израчунава отпорност R_2 непознатог отпорника.

Корак 1: Формирање кола Повезивање

Елемент	Опис
R1 (познате вредности нпр 1 kΩ)	Повезује се између 5V и A0
R2 (непознате вредности)	Повезује се између A0 и GND
A0	Спој R1 и R2

💡 Што је отпорност R_2 већа у односу на R_1 , већи ће бити напон на аналогном улазу



Корак 2: Arduino код

```
vconst int analogPin = A0;
const float Vin = 5.0;
const float R1 = 1000.0; // Known resistor in ohms

void setup() {
  Serial.begin(9600);
}

void loop() {
  int raw = analogRead(analogPin);
  float Vout = (raw * Vin) / 1023.0;
  float R2 = R1 * (Vin / Vout - 1);

  Serial.println(R2); // Send resistance to Processing
  delay(500);
}
```

Step 3: Processing код

```
import processing.serial.*;

Serial myPort;
String val = "";
float resistance = 0;

void setup() {
  size(400, 200);
  myPort = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
  background(255);
  fill(0);
  textSize(20);
  text("Measured Resistance:", 50, 80);
  text(nf(resistance, 1, 2) + " Ω", 50, 120);

  if (myPort.available() > 0) {
    val = myPort.readStringUntil('&apos;\n&apos;');
    if (val != null) {
      resistance = float(trim(val));
    }
  }
}
```

Како систем функционише

- Arduino читава напон у средишњој тачки напонског делиоца.
- На основу Омовог закона израчунава непознату отпорност.
- Израчуната вредност отпорности шаље се путем серијске комуникације у Processing окружење.
- Processing приказује вредност у реалном времену.

Задаци за ученике

Level	Task
● Основни	Приказати вредност отпорности као текст
● Средњи	Мењати боју текста у зависности од опсега отпорности
● Напредни	Направити визуелизацију у облику бројчаника или мерача
● Изазов	Сачувати више мерења и приказати их на графикону

Решавање проблема

Проблем	Могући узрок
Очитана вредност је 0 или бесконачност	R2 није повезан или је у кратком споју
Мерење није тачно	Вредност отпорника R1 је погрешно унета у програм
Вредности осцилирају	Лош контакт на breadboard плочи

Идеје за проширење пројекта

- аутоматска класификација отпорника према вредности отпорности;
- израда графичког мерача или аналогног инструмента;
- приказ бојног кода отпорника уз измерену вредност;
- мерење променљивих отпорника, као што су потенциометри;
- чување резултата мерења у CSV датотеку;
- израда прототипа једноставног дигиталног мултиметра;
- додавање подршке за OLED или LCD дисплеј;
 - израда режима за калибрацију ради повећања тачности мерења;
 - поређење измерених вредности са номиналним вредностима и толеранцијама отпорника. Creating a calibration mode for more accurate measurements

Закључак

Овај пројекат упознаје ученике са техникама електричних мерења, колима са напонским делиоцем и визуелизацијом података у реалном времену. Комбиновањем Arduino Uno платформе и Processing окружења ученици стичу практична искуства из електронике, програмирања и примењене физике кроз практичну STEM активност усмерену на мерење и анализу електричне отпорности.

Пројект 5: Аутоматски систем за наводњавање

Опис

У овом пројекту користи се капацитивни сензор влажности земљишта за мерење садржаја воде у земљишту. Ученици израђују аутоматски систем за наводњавање који у реалном времену прати стање земљишта и управља радом пумпе за воду. Подаци са сензора се путем серијске комуникације преносе на рачунар и визуелизују у Processing окружењу.

Пројекат повезује електронику, програмирање, физику и науке о животној средини и представља практичан увод у аутоматизацију, IoT (Internet of Things) системе и управљање засновано на сензорима.

Циљеви

Ученици ће:

- научити како функционишу капацитивни сензори влажности земљишта;
- повезати сензор и релејни модул са Arduino микроконтролером;
- очитавати и тумачити аналогне податке са сензора;
- користити серијску комуникацију за пренос података до рачунара;
- визуелизовати податке о влажности земљишта у реалном времену помоћу Processing окружења;
- управљати пумпом за воду преко релејног модула из Arduino и Processing окружења;
- изградити потпуно аутоматизован систем за наводњавање заснован на граничним вредностима влажности.

Резултати учења

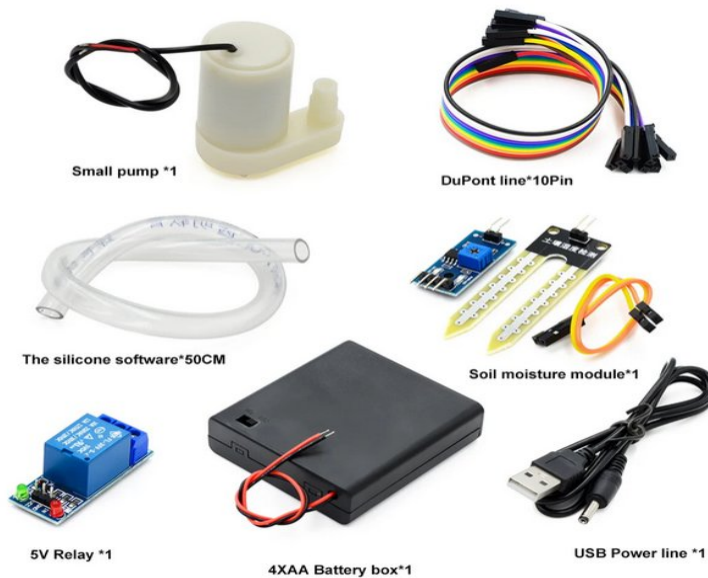
Након завршене радионице ученици ће бити у стању да:

- правилно повежу капацитивни сензор влажности земљишта и релејни модул са Arduino плочом;
- очитавају и скалирају аналогне вредности са сензора;
- разумеју и примењују логику управљања засновану на граничним вредностима;
- користе серијску комуникацију за размену података између Arduino и Processing окружења;
- визуелизују податке са сензора помоћу текста, графичких трака и статусних индикатора у Processing окружењу;
- шаљу управљачке команде из Processing окружења ка Arduino плочи;
- пројектују комплетан аутоматизован систем који повезује сензоре и извршне елементе.

Потребна опрема

Component	Quantity	Purpose
Arduino Uno	1	Микроконтролер
Капацитивни сензор за мерење влажности тла v1.2	1	Мерење влажности тла
Релеј	1	Укључивање/искључивање пумпе
Пумпа за воду	1	наводњавање
Спољашњи извор напајања (5V или 12V)	1	Напајање пумпе

Component	Quantity	Purpose
Силиконско црево (50 cm)	1	Довод воде
Breadboard + жице	1 комплет	Повезивање компоненти
USB cable	1	Повезивање са компјутером
Processing IDE	1	визуелизација



Теоријске основе

Како функционише сензор влажности земљишта?

Капацитивни сензор влажности земљишта даје аналогни напон чија се вредност смањује са повећањем влажности земљишта. Суво земљиште даје већу вредност (око 1023), док влажно земљиште даје мању вредност (око 300). За разлику од отпорничких сензора, капацитивни сензор не кородира током времена.

💡 💡 **Физичка основа:** Влажно земљиште има већу диелектричну пропустљивост, што мења капацитивност сензора и напон на његовом излазу.

Релеј модул

Релеј је електрично управљани прекидач. Мали сигнал од 5 V са Arduino плоче управља струјним колом пумпе које може радити на другом напону. Када је улаз релеја у стању HIGH, прекидач је најчешће отворен (пумпа је искључена), а када је у стању LOW, прекидач се затвара (пумпа је укључена). Увек проверите логику рада конкретног релејног модула који користите.

Серијска комуникација

Arduino шаље податке са сензора рачунару путем серијске комуникације преко USB порта. На пример:

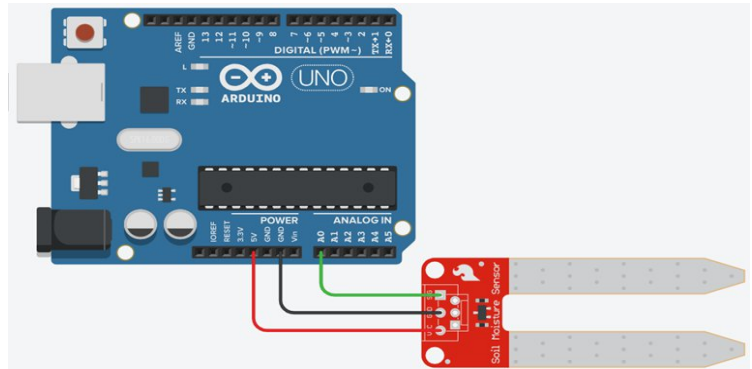
```
Serial.println(moisture);
```

Processing чита ову вредност и користи је за ажурирање визуелног приказа и одређивање статуса пумпе

Вежба 1: Прикупљање података и визуелизација

Wiring — Sensor

Пин сензора	Повезивање
VCC	3.3V или 5V
GND	GND
AOUT	Arduino пин A0



Arduino код

```
const int sensorPin = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int moisture = analogRead(sensorPin);
  // 0 (wet) to 1023 (dry)
  Serial.println(moisture);
  delay(300);
}
```

Сензор даје аналогни напон чија се вредност смањује са повећањем влажности земљишта.

Processing код — текстуални и графички приказ помоћу траке

```
import processing.serial.*;

Serial myPort;
String val = "";
int moisture = 0;

void setup() {
    size(400, 200);
    myPort = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
    background(255);
    fill(0);
    textSize(20);
    text("Soil Moisture: " + moisture, 50, 60);

    // Visual bar
    fill(0, 100, 255);
    rect(50, 100, map(moisture, 1023, 300, 0, 300), 30);
    // Adjust range as needed
}

void serialEvent(Serial p) {
    val = p.readStringUntil('&apos;\n&apos;');
    if (val != null) {
        val = trim(val);
        if (val.matches("\\d+")) {
            moisture = int(val);
        }
    }
}
```

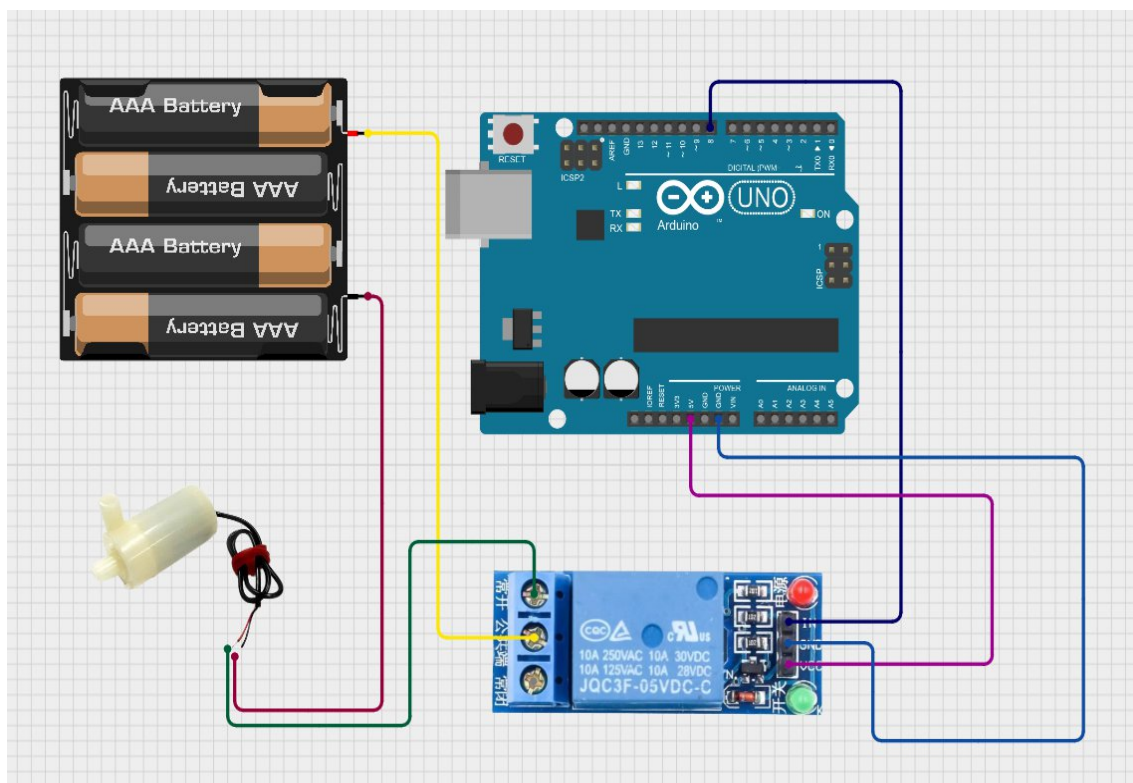
Вежба 2: Управљање пумпом за воду из Processing-аа

Додатне компоненте:

- релејни модул;
- пумпа за воду;
- спољашње напајање од 12 V (за пумпу)

Повезивање релеја и пумпе

Веза	Опис
Релеј IN	Arduino D8
Релеј VCC	Arduino 5V
Релеј GND	Arduino GND
Relay COM	Позитивни (+) прикључак спољашњег напајања
Релеј NO	Позитивни (+) прикључак пумпе
Негативни (–) прикључак пумпе	GND спољашњег напајања (– прикључак)



Arduino код

```
void setup() {
  Serial.begin(9600);
  pinMode(9, OUTPUT);
  digitalWrite(9, HIGH); // Ensure Pump starts OFF
}

void loop() {
  if (Serial.available() > 0) {
    while (Serial.available() > 1) { // Flush extra
bytes
      Serial.read();
    }

    char cmd = Serial.read();

    if (cmd == '1') {
      digitalWrite(9, LOW); // Pump ON
    } else if (cmd == '0') {
      digitalWrite(9, HIGH); // Pump OFF
    }
  }
}
```

Processing код

```
import processing.serial.*;

Serial myPort;

void setup() {
  size(300, 100);
  myPort = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
  background(200);
  fill(0);
  text("Press '1' to turn ON pump, '0' to turn
OFF", 20, 50);
}

void keyPressed() {
  if (key == '1' || key == '0') {
    myPort.write(key);
  }
}
```


Arduino код

```
#define RELAY_PIN 8
#define SENSOR_PIN A0
#define THRESHOLD 600

void setup() {
  Serial.begin(9600);
  pinMode(RELAY_PIN, OUTPUT);
}

void loop() {
  int moisture = analogRead(SENSOR_PIN);
  Serial.println(moisture);

  if (moisture > THRESHOLD) {
    digitalWrite(RELAY_PIN, HIGH); // Dry → Water ON
  } else {
    digitalWrite(RELAY_PIN, LOW);  // Wet → Water OFF
  }

  delay(1000);
}
```

Processing код — приказ стања

```

import processing.serial.*;

Serial myPort;
String val;

void setup() {
  size(400, 200);
  myPort = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
  if (myPort.available() > 0) {
    val = myPort.readStringUntil('&apos;\n&apos;');
    if (val != null) {
      int moisture = int(trim(val));
      background(255);
      fill(0);
      textSize(20);
      text("Soil Moisture: " + moisture, 50, 100);
      if (moisture > 600) {
        fill(255, 0, 0);
        text("Status: DRY - Watering", 50, 140);
      } else {
        fill(0, 150, 0);
        text("Status: WET - Idle", 50, 140);
      }
    }
  }
}

```

Задаци за ученике

Ниво	Задатак
● Основни	Приказати вредност влажности земљишта и текстуално статус система
● Средњи	Направити траку влажности означену бојама (плава = влажно, црвена = суво)
● Напредни	Додати дугме у Processing окружењу за ручно управљање пумпом
● Изазов	Бележити податке о влажности земљишта са временским ознакама у CSV датотеку

Решавање проблема

Проблем	Могући узрок
Пумпа се не укључује	Неправилно повезан релеј или неслагање броја пина у програму
Сензор стално показује максималну вредност	Сензор није у земљишту или је неисправан
Пумпа ради непрекидно	Праг је постављен превисоко, смањити вредност параметра THRESHOLD
Processing не приказује податке	Погрешан COM порт или не неусклађена брзина серијске комуникације

☑ **Савет:** Користите `Serial.list()` да бисте пронашли исправан COM порт. По потреби измените вредност `Serial.list()[0]`.

Практична примена

Аутоматски системи за наводњавање и праћење влажности земљишта користе се у:

- паметним баштама и системима кућне аутоматизације;
- пољопривредној производњи и прецизном наводњавању;
- системима за контролу услова у пластеницима и стакленицима;
- урбаној пољопривреди и вертикалним баштама;
- IoT платформама за праћење услова животне средине;
- истраживачким станицама за проучавање земљишта и биљака;
- системима за уштеду воде у сушним подручјима.

Међупредметна повезаност

Предмет	Веза
Физика	Капацитивност, аналогни напон, електрична кола, управљање релејем
Биологија и екологија	Биологија биљака, састав земљишта, кружење воде
Информатика	Програмирање, серијска комуникација, обрада података
Техника и технологија	Електроника, интеграција сензора, управљање извршним елементима
Математика	Скалирање података, граничне вредности, тумачење графикана
Одрживи развој	Очување воде, одржива пољопривреда, климатски услови

DigComp компетенције

Радионица развија следеће DigComp 2.2 компетенције:

DigComp област	Развијена компетенција
1. Информациона и податковна писменост	Прикупљање и тумачење података са сензора у реалном времену
2. Комуникација и сарадња	Представљање резултата и тимски рад
3. Креирање дигиталног садржаја	Програмирање у Arduino и Processing окружењима
4. Safety	Разумевање безбедне употребе електронских компоненти и релеја
5. Решавање проблема	Дијагностика хардверских проблема и отклањање грешака у програмима

Конкретно, радионица развија:

- дигиталну писменост и рачунарско размишљање;
- вештине решавања проблема и систематског отклањања грешака;
- прикупљање и анализу података у реалном времену;
- програмирање микроконтролера и визуелних корисничких интерфејса;
- сарадњу у практичном STEM окружењу.

Идеје за проширење пројекта

Ученици који успешно реализују све вежбе могу проширити пројекат:

- додавањем OLED дисплеја за директан приказ влажности земљишта;
- слањем података путем WiFi мреже коришћењем ESP8266 или ESP32 модула;
- израдом web контролне табле за удаљено праћење влажности земљишта;
- повезивањем са cloud платформама (нпр. ThingSpeak или Blynk) ради бележења података;
- израдом система за наводњавање са више зона за више саксија или парцела;
- додавањем графика влажност у реалном времену у Processing окружењу;
- интеграцијом временске прогнозе преко API сервиса како би се избегло наводњавање током кишних дана;
- додавањем резервног батеријског напајања и соларног панела за самосталан рад система.

Проширење	Тежина	Нове вештине
OLED диспеле	● Средњи	I2C комуникација, библиотеке за дисплеје
WiFi пренос података	● Напредни	ESP32/ESP8266, мрежно програмирање
Cloud контролна табла	● Напредни	Интеграција API сервиса, визуелизација података
Систем са више зона	● Изазов	Више сензора, логика управљања
Соларно напајање	● Изазов	Управљање напајањем, пројектовање хардвера

Закључак

Овај пројекат представља практичан увод у аутоматизацију, управљање засновано на сензорима и визуелизацију података у реалном времену. Комбиновањем Arduino и Processing окружења са сензором влажности земљишта и релејним модулом ученици стичу практична искуства из електронике, програмирања и праћења услова животне средине.

Систем за аутоматско наводњавање показује како једноставни дигитални алати могу решавати стварне проблеме — од уштеде воде у кућним баштама до управљања великим пољопривредним системима. Кроз три постепено сложеније вежбе ученици развијају знања и вештине које су непосредно повезане са развојем IoT система, аутоматизацијом и одрживим технологијама.

Хакатон и примена стечених знања

Поред вођених радионица, програм **Digital Challenge** обухвата и низ такмичарских изазова у форми хакатона, осмишљених да подстакну креативност, самостално решавање проблема, брзо прототипирање и практичну примену стечених STEM знања у реалистичним ситуацијама.

Ови изазови представљају прелазак са структурисаних вежби ка отвореним инжењерским и проблемским задацима. Од ученика се очекује да повежу знања из електронике, физике, програмирања, сензорских система, серијске комуникације, визуелизације података и аутоматизације како би у ограниченом времену развили функционално решење.

Хакатонски формат подстиче:

- примену стечених знања;
- сарадничко решавање проблема;
- инжењерски приступ пројектовању;
- креативност и иновативност;
- отклањање проблема у реалним условима;
- доношење одлука у реалном времену.

За разлику од класичних школских вежби, изазови симулирају стварне инжењерске ситуације у којима ученици треба да:

- самостално анализирају проблем;
- изаберу одговарајуће хардверске и софтверске алате;
- пројектују електронска кола;
- развију алгоритме;
- тестирају и отклањају грешке у систему;
- представе функционалан прототип.

Задаци на хакатону директно су повезани са темама обрађеним у оквиру радионица.

Област радионице	Изазови на хакатону
Разделници напона и аналогна мерења	Мерење непознате отпорности
LED диодне и електрична мерења	Одређивање напона укључења LED диоде
Светлосни сензори и обрада сигнала	Оптичка комуникација Морзевим кодом
Системи за мерење температуре	Детекција „невидљивих робота“ мерењем температуре
Ултразвучни сензори и визуелизација	Паркинг сензор и радарски систем
Визуелизација у Processing окружењу	Сви задаци са приказом података
Аутоматизација и системи са прагом реаговања	Нводњавање и логика управљања паркингом сензором

Градација изазова по сложености омогућава ученицима да напредују од:

- једноставних мерења;
- преко интерактивних система;
- до потпуно аутономних система и комуникационих пројеката.

Посебна пажња посвећује се:

- стабилности система;
- организацији програмског кода;
- поузданости мерења;
- јасноћи визуелизације;
- успешној демонстрацији функционалног прототипа.

Научнофантастични сценарији који се користе у појединим задацима (инвазија ванземаљаца, невидљиви роботи, сигурна светлосна комуникација) намерно су уведени ради повећања мотивације и ангажовања ученика, али истовремено захтевају примену стварних научних и инжењерских принципа.

Кроз ове активности ученици развијају практичне компетенције из области:

- интеграције сензора;
- програмирања уграђених система;
- серијске комуникације;

- обраде сигнала;
- визуелизације података;
- аутоматизације и управљања;
- развоја прототипова;
- тимског рада и презентације техничких решења.

Hackathon Challenges

Изазов 1.

Ваш задатак је да, користећи инсталирани софтвер, развојне системе и компоненте доступне на радном месту, одредите непознате вредности отпорности плавог, црвеног и зеленог отпорника.

10 поена

- 2 поена — исправно повезано коло
- 2 поена — одговарајући програм
- 3 поена — исправно одређена вредност плавог отпорника $R_1 = \underline{\hspace{2cm}}$
- 1.5 поена — исправно одређена вредност црвеног отпорника $R_2 = \underline{\hspace{2cm}}$
- 1.5 поена — исправно одређена вредност зеленог отпорника $R_3 = \underline{\hspace{2cm}}$

Изазов 2.

Користећи доступне компоненте, развојне системе и инсталирани софтвер, одредите напон укључења црвене, жуте и зелене LED диоде, односно напон на диоди при коме она почиње да емитује светлост. Струја кроз LED диоду не сме прећи 50 mA.

12 поена

- 4 поена — струја не прелази 50 mA
- 3 поена — исправан метод мерења
- 2 поена — вредност за црвену LED $U_1 = \underline{\hspace{2cm}}$
- 1 поена — вредност за жуту LED $U_2 = \underline{\hspace{2cm}}$
- 1 поена — вредност за зелену LED $U_3 = \underline{\hspace{2cm}}$
- 1 поена — објашњење и презентација мерења

Изазов 3.

У блиској будућности догодила се инвазија ванземаљаца. Једини комуникациони канал који не могу да контролишу јесте комуникација заснована на светлости. Ваш задатак је да пошаљете општи позив, **CQ**, Морзеовим кодом свима који могу да виде ваш сигнал. Оптимална струја за рад LED диоде која се користи за пренос сигнала износи 15 mA. Тачка траје 200 ms. Користећи инсталирани софтвер, развојне системе и компоненте доступне на радном месту, решите овај проблем.

15 поена

- 3 поена — исправно одређивање вредности отпорника
- 2 поена — исправно повезано коло
- 4 поена — исправан Морзеов код за C
- 4 поена — исправан Морзеов код за Q
- 2 поена — исправан размак између слова и понављање слова

Изазов 4.

Ванземаљци имају невидљиве роботе који се могу открити само мерењем температуре у

просторији. Ако температура, без видљивог спољашњег узрока, пређе 30 °C, постоји могућност да се у близини налази невидљиви робот.

Измерите температуру у учионици користећи доступне компоненте, развојне системе и инсталирани софтвер.

а) Систем мора да ради независно од рачунара и да обезбеди светлосну сигнализацију када температура пређе 30 °C.

б) Прикажи измерене вредности користећи Processing.

а) 10 поена • 2 поена — исправно повезан сензор • 3 поена — тачно мерење температуре • 3 поена — исправан одговор на 30°C • 2 поена — стабилно функционисање без компјутера	б) 10 поена • 3 поена — серијски пренос података • 3 поена — тачан приказ вредности • 2 поена — мерења у реалном времену • 2 поена — јасноћа приказа
--	---

Изазов 5.

Због све већих саобраћајних гужви, возила је потребно паркирати што ближе једно другом. Зато су потребни паркинг сензори који показују растојање између возила и препреке.

Ваш задатак је да, користећи доступне ресурсе, осмислите једноставан паркинг сензор који близину возила приказује помоћу зелене и црвене LED диоде.

Зелена LED диода светли пуним интензитетом све док је растојање између возила и препреке веће од 15 cm. Њен интензитет се смањује како се возило приближава препреци. За растојања мања од 10 cm укључује се црвена LED диода, а њен интензитет расте до максимума како се растојање смањује до 5 cm. За растојања мања од 5 cm црвена LED диода почиње брзо да трепери.

а) Систем мора да ради независно од рачунара.

б) Осмислите сопствени начин приказа растојања и представите га на рачунару помоћу Processing окружења.

а) 15 поена • 3 поена — тачно мерење растојањат • 3 поена — зелена LED се укључује ако је растојање >15 cm • 3 поена — зелена LED слабије светли како се препрека приближава • 3 поена — црвена LED појачава светлост ако је растојање 10–5 cm • 3 поена — црвена LED брзо трепери за растојање <5 cm	б) 10 поена • 4 поена — растојање се приказује у Processing-у • 3 поена — логична и јасна визуелизација • 3 поена — оригиналност приказа
---	--

Специјални изазов.

Једини безбедан комуникациони канал током инвазије ванземаљаца јесте светлост.

Морзеов код и првих пет слова абецедe користе се за слање кодираних порука помоћу светлосних сигнала.

Тачка у Морзеовом коду траје 200 ms. Користећи инсталирани софтвер, развојне системе и компоненте доступне на радном месту, направите прототип уређаја који ће приказати примљену светлосну поруку.

За декодирање користите табелу:

Code	Symbol	Code	Symbol	Code	Symbol
AAA	A	ADA	M	BAC	Y
AAB	B	ADB	N	BAD	Z
AAC	C	ADC	O	EAA	0
AAE	D	ADD	P	EAB	1
ABA	E	ADE	Q	EAC	2
ABC	F	AEA	R	EAD	3
ABD	G	AEB	S	EAE	4
ABE	H	AEC	T	EBA	5
ACA	I	AED	U	EBB	6
ACB	J	AEE	V	EBC	7
ACD	K	BAA	W	EBD	8
ACE	L	BAB	X	EBE	9

50 поена

- 4 поена — исправна детекција светлосних импулса
- 4 поена — исправно разликовање кратких и дугих сигнала
- 10 поена — исправно препознавање слова
- 10 поена — исправно препознавање трословних кодова
- 5 поена — стабилност система
- 7 поена — организација програмског кода
- 10 поена — успешна демонстрација рада на тест примеру

Хакатон изазови — 2 дана

Digital Challenge — практични задаци за такмичаре

Изазов 1: Мерење растојања помоћу ESP32 и HC-SR04

Digital Challenge — практични задаци за такмичаре

Циљ: Повезати ултразвучни сензор на ESP32, измерити растојање и визуелизовати податке.

Укупно поена: 35

Задатак 1 — Нацртати шему [5 бод.]

Користећи Circuit Designer (<https://app.circuitdesigner.com/>), нацртати шему повезивања ESP32 плоче са HC-SR04 ултразвучним сензором.

- Јасно означити све пинове: VCC, GND, TRIG, ECHO.
- Известити (export) или снимити екран и приложити у фолдер за предају.

💡 ESP32 ради на 3.3 V логици; ECHO пин HC-SR04 сензора даје излаз од 5 V — по потреби користити разделник напона (voltage divider) или level shifter.

Задатак 2 — Хардверско повезивање [5 бод.]

Физички повезати ESP32 и HC-SR04 на бредboard-у, пратећи нацртану шему.

- Двоструко проверити VCC (5 V или 3.3 V) и GND пре укључивања напајања.
- Оцењивач ће прегледати везе на бредboard-у.

Задатак 3 — Програм у Arduino IDE-у [10 бод.]

Написати C++ програм (Arduino скицу) који:

- Шаље окидачки импулс (trigger pulse) од 10 μ s на TRIG пин.
- Мери трајање одјека на ECHO пину користећи функцију pulseIn().
- Израчунава растојање у сантиметрима (брзина звука = 0.034 cm/ μ s).
- Исписује растојање у Serial Monitor сваких 500 ms.

Тестирати на најмање два различита растојања и упоредити резултат са лењиром.

Задатак 4 — Верификација [5 бод.]

Отворити Serial Monitor (9600 бода). Снимити екран (screenshot) који приказује:

- Најмање 5 узастопних читавања.
- Две јасно различите вредности растојања.

Задатак 5 — Визуелизација у Processing-у [10 бод.]

Написати Processing скицу (Java режим) која:

- Чита вредности растојања са серијског порта.
- Приказује нумеричку вредност растојања на екрану.
- Исцртава попуњени правоугаоник чија је ширина пропорционална измереном растојању.

💡 Користити *Serial* библиотеку у *Processing*-у. Ускладити брзину преноса (*baud rate*) са *Arduino* скицом (9600).

Изазов 2: Управљање серво мотором помоћу растојања

Циљ: Користити очитавање ултразвучног сензора за управљање позицијом серво мотора.

Укупно поена: 30

Задатак 1 — Нацртати шему [5 бод.]

У *Circuit Designer*-у нацртати везу између *ESP32*, *SG90* серво мотора и *HC-SR04* сензора.

- Јасно означити све везе.
- Приказати везу сигналног пина серва.

Задатак 2 — Хардверско повезивање [10 бод.]

Повезати *ESP32*, *SG90* серво и *HC-SR04* на бредборд.

- Напајати серво са спољашње 5V шине (или USB-a), А НЕ директно са *ESP32* GPIO пина.
- Потврдити стабилан рад пре програмирања.

💡 *SG90* серво повлачи до 250 mA струје — користити спољашњи извор напајања како би се избегло ресетовање *ESP32*.

Задатак 3 — Програм у *Arduino IDE*-у [10 бод.]

Написати C++ програм који:

- Непрекидно чита растојање са *HC-SR04* сензора.
- Ротира серво на 0° (лево) када је растојање мање од 10 cm.
- Ротира серво на 180° (десно) када је растојање веће од 20 cm.
- Задржава серво на 90° (средина) за растојања између 10–20 cm.
- Исписује растојање и угао серва у *Serial Monitor*.

Задатак 4 — Верификација [5 бод.]

Демонстрирати да серво исправно реагује на три опсега растојања. Оцењивач ће држати предмет на различитим растојањима током оцењивања.

Изазов 3: Сонарни скенер са визуелизацијом у *Processing*-у

Циљ: Направити ротирајући сонарни систем и приказати радарски приказ у Processing-у.

Укупно поена: 50

Задатак 1 — Нацртати шему [5 бод.]

У Circuit Designer-у нацртати комплетну везу (ESP32 + SG90 + HC-SR04).

- Сензор и серво треба приказати монтиране заједно на носачу.

Задатак 2 — Хардверско повезивање [10 бод.]

Физички саставити сонарну јединицу:

- Монтирати HC-SR04 на врх руке SG90 серва тако да заокружује лево/десно.
- Повезати каблове за напајање и сигнал без ограничавања ротације.

Задатак 3 — Програм у Arduino IDE-у [15 бод.]

Написати C++ програм који:

- Заокружује серво од 0° до 180° и назад у корацима од 1°.
- Мери растојање на сваком углу.
- Шаље податке преко Serial-а у формату: ANGLE,DISTANCE\n
- Укључује кратко кашњење (≤ 30 ms по кораку) како би се читавања сензора стабилизовала.

Задатак 4 — Верификација [5 бод.]

Отворити Serial Monitor. Потврдити да вредности угла заокружују 0–180 и да су вредности растојања смислене.

Задатак 5 — Визуелизација у Processing-у [15 бод.]

Написати Processing скицу која:

- Чита линије ANGLE,DISTANCE са серијског порта.
- Приказује радарски приказ: зелена линија заокруживања на црној позадини.
- Исцртава тачку или лук на измереном растојању за сваки угао.
- Додаје ознаке угла и растојања око приказа.

💡 Користити конверзију поларних у Декартове координате: $x = dist * \cos(angle)$, $y = dist * \sin(angle)$.

Изазов 4 (напредни): WiFi сонар са Android Processing апликацијом

Циљ: Пренети сонарне податке преко WiFi-ја (UDP) и визуелизовати их у Android Processing апликацији.

Укупно поена: 30

Бонус изазов — покушати тек након завршетка Изазова 1–3.

Задатак 1 — Програм у Arduino IDE-у [15 бод.]

Написати C++ програм који:

- Повезује ESP32 на познату WiFi мрежу (SSID/лозинка у коду).
- Заокружује серво 0°–180° и мери растојање на сваком кораку.
- Шаље ANGLE,DISTANCE као UDP пакет на назначену Android IP адресу и порт.
- Непрекидно понавља заокруживање.

💡 Користити WiFiUDP библиотеку на ESP32. Потврдити да су ESP32 и Android уређај на истој мрежи.

Задатак 2 — Визуелизација у Android Processing-у [15 бод.]

Написати Processing for Android скицу која:

- Ослушкује UDP пакете на назначеном порту.
- Рашчлањује ANGLE,DISTANCE из сваког пакета.
- Приказује потпуни радарски приказ са линијом заокруживања и тачкама растојања.
- Ради глатко при ≥ 20 fps на Android уређају.

💡 Користити *ketai* или *processing.net UDP* библиотеку за Android режим. По могућству тестирати на правом уређају.

Педагошке и методичке основе

Увод

Скуп радионица **Digital Challenge** осмишљен је као практичан STEM образовни програм који кроз пројектно оријентисано учење повезује електронику, програмирање, физику, инжењерство, визуелизацију података и решавање проблема из стварног живота.

Радионице су засноване на практичном експерименталном раду са микроконтролерима, сензорима, актуаторима и алатима за визуелизацију података, омогућавајући ученицима да кроз активно учешће и тимски рад развијају техничке и трансверзалне компетенције.

Образовни приступ подстиче:

- учење кроз рад;
- истраживачко учење;
- експериментисање;
- интердисциплинарно решавање проблема;
- креативност и иновативност.

Материјали су погодни за:

- формално образовање;
- ваннаставне STEM активности;
- makerspace окружења;
- хакатоне;
- роботичке секције;
- неформалне облике учења;
- активности стручног усавршавања наставника.

Образовна филозофија

Скуп радионица заснива се на идеји да ученици најефикасније уче када активно пројектују, израђују, тестирају, анализирају и унапређују реалне системе.

Уместо да се фокусирају искључиво на теоријска објашњења, ученици

- раде са стварним уређајима;
- прикупљају и тумаче податке из реалног окружења;
- развијају функционалне прототипове;
- визуелизују информације;
- решавају аутентичне инжењерске проблеме.

Методологија подстиче:

- експериментисање;
- итеративни процес пројектовања;
- отклањање грешака и проблема;
- тимски рад;
- рефлексивно учење.

Посебан нагласак стављен је на повезивање научних концепата са практичним применама у свакодневном животу и савременим технологијама.

Методолошки приступ

Радионице комбинују више комплементарних образовних приступа.

Вршњачко и учење кроз сарадњу

Радионице су осмишљене тако да подржавају:

- тимски рад;
- сарадничко решавање проблема;
- вршњачко менторство;
- размену знања;
- представљање резултата пројекта.

Ученици се подстичу да заједнички разматрају решења, упоређују различите приступе и сараднички отклањају проблеме у хардверским и софтверским системима.

Учење засновано на пројектима (Project-Based Learning - PBL)

Ученици уче кроз развој комплетних функционалних пројекта, као што су:

- системи за праћење услова животне средине;
- радарске симулације;
- паметни системи за наводњавање;
- аутоматизовани сензорски системи;
- интерактивни визуелни интерфејси.

Пројекти се развијају од основне функционалности ка напреднијим проширењима и отвореним изазовима.

Учење кроз истраживање

Ученици се подстичу да:

- истражују понашање сензора;
- проверавају хипотезе;
- упоређују резултате мерења;
- калибришу системе;
- идентификују грешке;
- унапређују перформансе кроз експериментални рад.

Радионице подстичу научни начин размишљања и закључивање засновано на доказима.

Учење засновано на изазовима

Свака радионица обухвата:

- постепено сложеније вежбе;
- практичне задатке;
- ситуације за отклањање проблема;
- напредне изазове.

Оваква структура омогућава успешну реализацију активности ученицима различитих нивоа предзнања, истовремено пружајући могућност дубљег истраживања и напредовања.

Интеграција STEM области

Скуп радионица обједињује више STEM дисциплина у јединственом практичном оквиру.

Област	Примери
Физика	Простирање звука, напон, отпорност, капацитивност, сензори, мерење
Електроника	Електронска кола, напонски делиоци, релеји, извршни елементи
Информатика	Програмирање, серијска комуникација, обрада података
Инжењерство	Интеграција система, аутоматизација, израда прототипова
Математика	Анализа података, скалирање, графички приказ
Науке о животној средини	Системи за наводњавање, одрживост, праћење стања животне средине

Интердисциплинарна структура помаже ученицима да разумеју како су научни и технолошки концепти међусобно повезани у реалним системима..

DigComp и дигиталне компетенције

The workshops support the development of competencies aligned with the DigComp 2.2 framework, including:

DigComp област	Развијање компетенција
Информатичка и податковна писменост	Прикупљање, тумачење и визуелизација података са сензора
Комуникација и сарадња	Тимски рад и представљање резултата пројекта
Креирање дигиталног садржаја	Програмирање микроконтролера и визуелних интерфејса
Безбедност	Безбедан рад са електронским компонентама и системима
Решавање проблема	Отклањање грешака у хардверским и софтверским системима

Поред тога, радионице развијају:

- рачунарско размишљање;
- алгоритамско размишљање;
- дигиталну писменост;
- креативност;
- критичко мишљење;
- вештине инжењерског пројектовања.

Вештине 21. века

Активности у оквиру радионица подстичу развој:

- критичког мишљења;
- креативности;
- сарадње;
- комуникације;
- прилагодљивости;
- иницијативности;
- решавања проблема;
- технолошке писмености.

Ученици истовремено развијају техничке компетенције и трансверзалне вештине које су значајне за будуће STEM образовање и професионални развој.

Методологија вредновања

Вредновање може обухватати:

- успешну реализацију хардверског решења;
- функционалност програмског кода;
- процес решавања проблема;
- способност отклањања грешака;
- тумачење и анализу података;
- креативност предложених решења;
- тимски рад и сарадњу;
- презентацију пројекта;
- квалитет документације.
- Радионице подржавају:
- формативно вредновање током реализације активности;
- сумативно вредновање кроз завршне пројекте и демонстрације.

Инклузивност и приступачност

Структура радионица подржава инклузивно STEM образовање кроз:

- постепено повећање сложености задатака;
- активности прилагођене почетницима и напредним ученицима;
- сараднички рад;
- вршњачку подршку;
- визуелне приступе учењу;
- практичне активности засноване на непосредном искуству.

Материјали су осмишљени тако да подстакну учешће ученика различитих нивоа предзнања и омогуће шире укључивање у STEM области.

Посебна пажња може се посветити:

- подстицању учешћа девојчица у STEM областима;
- подршци недовољно заступљеним групама;
- прилагођавању активности различитим образовним контекстима.

Безбедносне смернице

Ученици треба да:

- искључе напајање пре измена у електронском колу;
- избегавају кратке спојеве;
- држе течности даље од електронских компоненти;
- пажљиво рукују сензорима и извршним елементима;
- провере исправност повезивања пре укључивања система.

Пројекти који укључују релеје, пумпе или спољашње изворе напајања захтевају додатни надзор и повећану пажњу у погледу електричне безбедности.

Рефлексивна и евалуација

Активности рефлексивне укључене су како би се подстакло дубље разумевање:

- понашања система;
 - инжењерских одлука;
 - поступака отклањања проблема;
 - примене решења у реалним условима.
- Повезаност са реалним светом

Примери питања за рефлексивну:

- Који је био највећи технички изазов?
- На који начин би систем могао бити унапређен?
- Који реалан проблем овај пројекат решава?
- Колико су мерења поуздана?
- Који додатни сензори или функције би могли бити додати?

Повезаност са реалним светом

Збирка радионица показује како се једноставни дигитални системи могу применити у областима као што су:

- праћење стања животне средине;
- паметна пољопривреда;
- аутоматизација;
- роботика;
- паметне куће;
- IoT системи;
- индустријски надзор;
- интерактивна визуелизација података.

Ученици стичу искуства са технологијама и концептима који су непосредно повезани са савременим инжењерством и дигиталним системима.

Примена и прилагођавање

Радионице су осмишљене тако да буду:

- нискобуџетне;
- модуларне;
- скалабилне;
- лако поновљиве у различитим окружењима.

Могу се прилагодити за:

- кратке радионице;
- полугодишне пројекте;
- хакатоне;
- STEM кампове;
- редовну наставу;
- онлајн или хибридне облике наставе.

Већина активности заснива се на широко доступним хардверским компонентама и софтверским алатима отвореног кода.