Universitatea de Vest din Timișoara

Smart::Bits

INFO

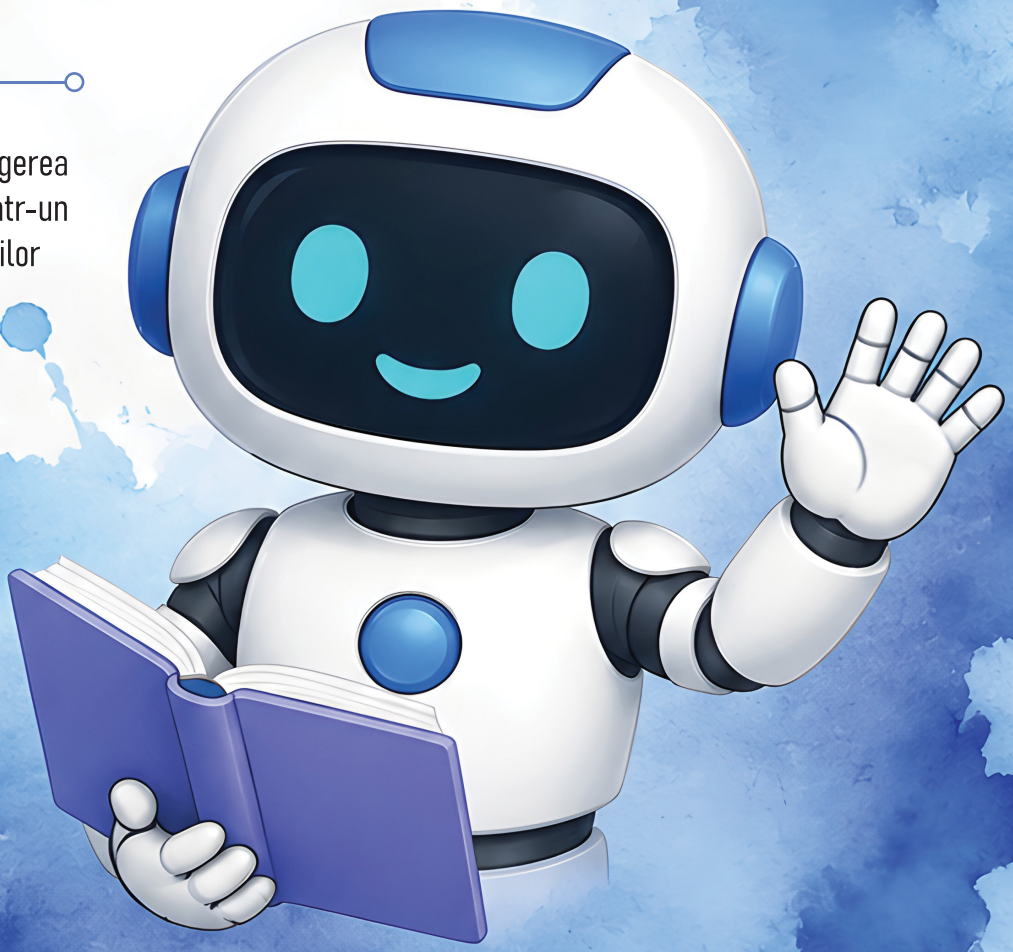
În parteneriat cu
MINISTERUL EDUCAȚIEI ȘI CERCETĂRII

ROBOTICA

PE ÎNȚELESUL TUTUROR

CLASELE IX-XII

Manual introductiv pentru înțelegerea principiilor de bază ale roboticii, într-un mediu virtual, accesibil tuturor elevilor



Theodor-Radu Grumeza | Matei Albu | Bogdan-Vasile Bozga | Marijus Musta

EUT

Universitatea de Vest din Timișoara

Smart::Bits

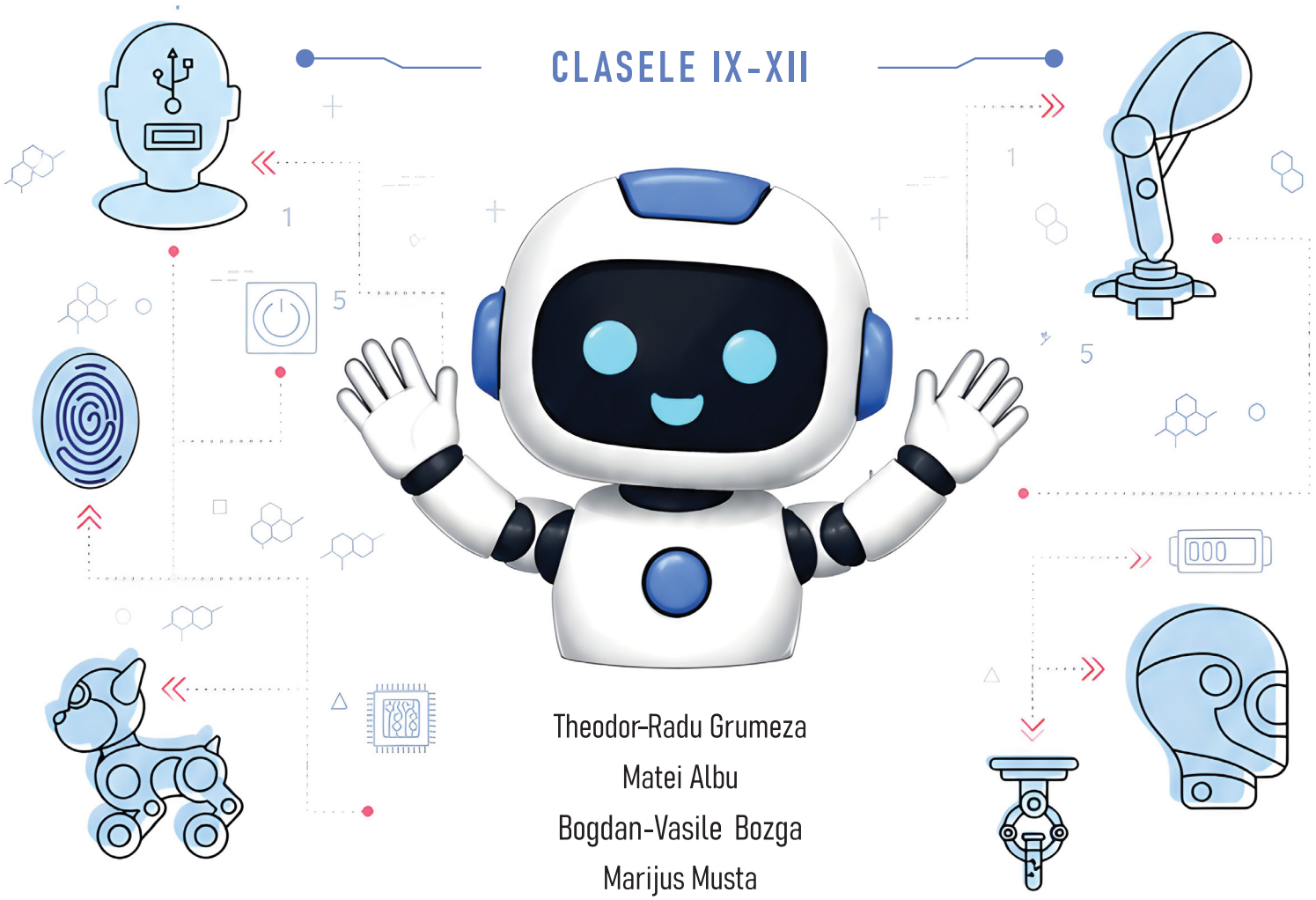
INFO

În parteneriat cu
MINISTERUL EDUCAȚIEI ȘI CERCETĂRII

ROBOTICA

PE ÎNȚELESUL TUTUROR

CLASELE IX-XII



Theodor-Radu Grumeza
Matei Albu
Bogdan-Vasile Bozga
Marijus Musta

EUV

Mențiuni privind platformele și mărcile

În activitățile propuse sunt utilizate platforme și aplicații aparținând unor terți. Denumirile și legăturile următoare sunt incluse pentru identificare, documentare și reproducerea corectă a etapelor didactice.

Platforme și resurse

- **Autodesk® Tinkercad®** — platformă online de proiectare, simulare de circuite electronice și modelare 3D, dezvoltată de Autodesk, Inc. Disponibilă la: tinkercad.com.
- **Bambu Studio (Bambu Lab)** — program de feliere utilizat pentru pregătirea modelelor în vederea imprimării 3D; lucrarea include capturi ale interfeței și un scurt fragment de G-code generat cu ajutorul aplicației. Disponibil la: bambulab.com/en/download/studio.
- **GrandpaCAD – STL Viewer** — vizualizator online pentru inspectarea fișierelor STL și a altor formate 3D. Disponibil la: grandpacad.com/en/tools/stl-viewer.
- **Kahoot!** — platformă de învățare bazată pe joc și chestionare interactive, menționată ca resursă de recapitulare la finalul lecțiilor. Disponibilă la: kahoot.com.

Notă privind fragmentul de G-code. Fragmentul este prezentat exclusiv pentru explicarea structurii unui fișier și a fluxului de pregătire a imprimării. El reprezintă un rezultat generat în contextul activității, nu un extras din codul-sursă al aplicației Bambu Studio. Parametrii trebuie verificați înaintea utilizării pe un echipament real.

Notă privind mărcile. Autodesk și Tinkercad sunt mărci înregistrate sau mărci comerciale ale Autodesk, Inc. și/sau ale filialelor ori afiliaților săi. Bambu Lab și Bambu Studio sunt mărci ale Shenzhen Tuozhu Technology Co., Ltd. Marca verbală micro:bit aparține Micro:bit Educational Foundation. Arduino este marcă sau marcă înregistrată a Arduino S.r.l. Kahoot! este marcă a Kahoot! AS. Denumirea GrandpaCAD și elementele grafice ale platformei aparțin titularului acesteia. Aceste denumiri sunt folosite exclusiv descriptiv; autorii și editura nu sunt afiliați titularilor, iar lucrarea nu este sponsorizată, aprobată sau susținută de aceștia.

Capturi de ecran

Notă generală. Toate capturile de ecran reproduse în prezenta lucrare redau interfețele platformelor menționate, în forma disponibilă la data realizării capturilor, și sunt utilizate exclusiv în scop didactic și ilustrativ. Drepturile asupra interfețelor și a elementelor grafice aparțin titularilor acestora.

PREFAȚĂ · ÎNAINTE DE PRIMA LECȚIE

Bine ai venit în atelier

Ai în față un manual care nu se citește, ci se lucrează. Fiecare noțiune din paginile lui ajunge, mai devreme sau mai târziu, într-un circuit construit de tine și într-un program scris de tine.

5 MODULE · 30 DE LECȚII · UN PROIECT FINAL AL ECHIPEI TALE

Robotica pare, de la distanță, o disciplină rezervată inițiaților. De aproape, ea este o înlănțuire de idei simple, așezate una peste alta: un curent care circulă printr-un circuit închis, o componentă care transformă energia electrică în mișcare, un senzor care traduce lumea în numere și un program care ia o decizie pe baza acelor numere. Manualul urmează exact acest drum, în ordinea în care ideile se sprijină una pe cealaltă.

Vei lucra în Tinkercad, un simulator gratuit în care componentele nu se ard și greșelile nu costă nimic. Fiecare lecție are două trasee, micro:bit și Arduino Uno: alegi unul și îl urmezi, iar la final compari cele două plăci, pentru că alegerea plăcii potrivite face parte din meseria de inginer.

Simulatorul este o alegere de principiu, nu o comoditate. Manualul acesta este construit astfel încât fiecare lecție să poată fi parcursă integral cu o tabletă și o conexiune la internet, fără nicio componentă cumpărată. Toate jocurile au și o variantă fără ecrane, iar nimic din ceea ce se evaluează nu depinde de o dotare pe care școala ta ar putea să nu o aibă. Robotica se învață la fel de bine pe un ecran: noțiunile sunt aceleași, iar deprinderile de gândire, cu atât mai mult.

Ce vei ști să faci la finalul manualului

- Să citești și să construiești un circuit electric, cu componentele potrivite și cu protecțiile lor
- Să comanzi motoare și servomotoare din program: sensul, viteza și poziția
- Să conectezi senzori de lumină, temperatură, distanță și mișcare, transformând citirile lor în decizii
- Să scrii programe cu condiții, praguri și mai multe ramuri, atât în blocuri, cât și în text
- Să proiectezi o piesă în 3D și să o pregătești pentru imprimare
- Să duci un proiect de echipă de la idee la prezentare, cu documentație tehnică

Modulul 1 · Fundamente	Curentul electric, circuitul închis, legea lui Ohm, LED-ul și primul program.
Modulul 2 · Motoare și actuatori	Motorul de curent continuu, reductoarele, tranzistorul, puntea H, controlul vitezei, servomotorul.
Modulul 3 · Senzori și achiziția datelor	Intrări digitale și analogice, convertorul, pragul, temperatura, distanța, fuziunea senzorilor.
Modulul 4 · Robotul complet și 3D	Automatizări cap-coadă, apoi imprimarea 3D, proiectarea în Tinkercad 3D și drumul de la model la obiect.
Modulul 5 · Proiectul final	Un sistem gândit de echipa ta, cu documentație tehnică și prezentare în fața unui juriu.

Trei lucruri de știut înainte de prima lecție

- ✓ Nimic nu se învață doar citind: deschide simulatorul și construiește pe măsură ce citești.
- ✓ Ce nu funcționează din prima nu este un eșec, ci o informație. Fiecare lecție are o secțiune de depanare, scrisă exact pentru acele momente.
- ✓ Ține un caiet alături: măsurătorile, pragurile și schemele notate acum îți vor folosi în proiectul final.

PRIMA LECȚIE ÎNCEPE CU O ÎNTREBARE SIMPLĂ: CE ESTE, DE FAPT, UN ROBOT?

Cuprinsul manualului

MODULUL 1 · 6 LECȚII

Fundamente: curent, circuite și primul program

1.1 **Ce este un robot? Prima ta simulare**
Simte, decide, acționează · plăcile micro:bit și Arduino Uno · primul circuit pornit în simulator

1.2 **Curentul electric și circuitul închis**
Analogia cu apa · tensiune, intensitate, rezistență · de ce circuitul trebuie să fie închis

1.3 **LED-ul și rezistorul de protecție**
Anod și catod · de ce LED-ul nu se conectează singur · pinii celor două plăci

1.4 **Legea lui Ohm: alegem rezistorul potrivit**
Formula U = I × R · calculul rezistorului · valoarea comercială din seria E12

1.5 **Primul circuit adevărat**
Montajul în serie · cablarea pas cu pas · circuitul care luminează, fără nicio linie de cod

1.6 **Primul tău program: LED-ul care clipește**
Pinul comandat din program · bucla care se repetă · aceleași instrucțiuni în blocuri și în Python

MODULUL 2 · 6 LECȚII

Motoare și actuatori: curentul pune lucrurile în mișcare

2.1 **Motorul de curent continuu: prima mișcare**
Ce este un actuator · cum produce motorul rotația · sensul dat de polaritate

2.2 **Roți dințate și reductoare: viteză sau forță**
Turație și cuplu · raportul de transmisie · măsurători în simulator

2.3 **Tranzistorul: motorul comandat din program**
De ce motorul nu se conectează direct la pin · tranzistorul și dioda de protecție · primul cod al modulului

2.4 **Schimbarea sensului de rotație: puntea H**
Cele patru comutatoare · circuitul integrat L293D · înainte și înapoi din cod

2.5 **Controlul vitezei: semnalul PWM**
Impulsuri și factor de umplere · trepte de viteză · scări diferite pe cele două plăci

2.6 **Servomotorul și primul mecanism: bariera de parcare**
Poziție comandată în grade · motorul pas cu pas · mini-proiectul modulului

MODULUL 3 · 6 LECȚII

Senzori și achiziția datelor: robotul percepe mediul

3.1 **Robotul ascultă: intrări digitale și prima decizie**
Pinii ca intrări · butonul și rezistorul de referință · instrucțiunea „dacă… atunci… altfel”

3.2 **Semnale care variază: potențiometrul și convertorul analog-digital**
Analogic și digital · convertorul pe 10 biți și scara 0–1023 · afișarea valorilor

3.3 **Lumina: fotorezistorul și noțiunea de prag**
Împărțitorul de tensiune · pragul măsurat, nu ghicit · lampa care se aprinde singură

3.4 **Temperatura: senzorul TMP36 și citirile cu zgomot**
Din număr în grade · variabilele și calibrarea · media care netezește zgomotul

3.5 **Distanța și mișcarea: senzorul ultrasonic și senzorul PIR**
Principiul ecoului · senzorul de parcare pe praguri · alarma de mișcare

3.6 **Senzorii roboților adevărați și fuziunea senzorilor**
Atlasul senzorilor: cameră, LiDAR, accelerometru, GPS · operatorul ȘI: doi senzori, o decizie

MODULUL 4 · 6 LECȚII

Primul robot complet și proiectarea 3D

4.1 **Bariera automată de parcare: primul robot complet**
Schema-bloc și pseudocodul · senzor, decizie și servomotor, cap-coadă

4.2 **Ventilatorul cu termostat**
„Altfel dacă”: decizia în trepte · temperatura comandă viteza motorului · alimentarea separată

4.3 **Lampa inteligentă și comanda manuală**
Operatorul SAU · trei intrări, o lampă · tabelul de adevăr și turul automatizărilor

4.4 **Cum funcționează o imprimantă 3D**
Fabricarea aditivă · imprimanta FDM și filamentul PLA · calculul numărului de straturi

4.5 **Primii pași în Tinkercad 3D**
Planul de lucru în milimetri · forme, găuri și grupare · brelocul cu numele tău

4.6 **De la model la obiect: STL și slicing**
Fișierul STL · slicerul și G-code-ul · înălțimea stratului, umplerea, suporturile

MODULUL 5 · 5 SECȚIUNI

Proiectul final: de la idee la prezentare

01 **Proiectul vostru**
Testul celor cinci criterii · douăsprezece puncte de plecare · echipa și cele cinci roluri

02 **Parcursul de lucru**
Patru pași, fiecare cu bifa lui · jurnalul de proiect · cele două capcane frecvente

03 **Documentația tehnică**
Ce conține și de ce contează · cele zece capitole · responsabil și termen pentru fiecare

04 **Prezentarea proiectului**
Scheletul de zece slide-uri · regulile prezentării · planul B pentru demonstrația live

05 **Ziua demo și evaluarea**
Desfășurarea · grila publică de 100 de puncte · cele cinci întrebări clasice ale juriului

Materiale care însoțesc lecțiile

Anexa elevului, la finalul fiecărui modul: răspunsurile la cele trei întrebări rapide ale fiecărei lecții.

Anexa de producție, pentru profesor: circuitele de pregătit în Tinkercad, linkurile, codurile QR și verificările dinaintea tipăririi.

Fișele de joc și testele Kahoot, câte unul pentru fiecare lecție, în volumul de anexe.

Codurile copiabile, accesibile prin codurile QR din lecții, pentru variantele Python și C++.

Modulul 1

Fundamente ale ROBOTICII

LECȚIA 1.1

Ce este un robot? Prima ta simulare.

În această lecție vei afla definiția robotului, vei identifica cele trei funcții pe care acesta le îndeplinește, vei studia plăcile de dezvoltare Micro:bit și Arduino Uno și vei rula prima ta simulare în mediul Tinkercad.

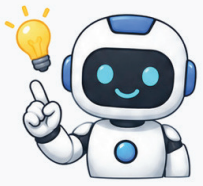
CE VEI ȘTI LA FINALUL LECȚIEI

- Să enunți definiția robotului și să explici cele trei funcții pe care acesta le îndeplinește: percepția, decizia și acțiunea.
- Să descrii principalele componente ale plăcilor de dezvoltare Micro:bit și Arduino Uno.
- Să identifici dispozitivele din viața de zi cu zi care corespund definiției robotului.
- Să pornești și să oprești o simulare în mediul online Tinkercad.

SECȚIUNEA 01 · TEORIE

Ce este, de fapt, un robot?

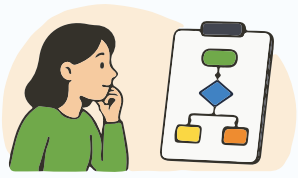
La auzul cuvântului „robot”, mulți oameni se gândesc la un personaj metalic din filme. În realitate, robotica operează cu o definiție precisă: **un robot este un sistem tehnic programabil care colectează informații din mediul înconjurător, le prelucrează conform unui program și acționează asupra mediului.** Orice robot îndeplinește, așadar, trei funcții, în această ordine:



1 SIMTE
percepție

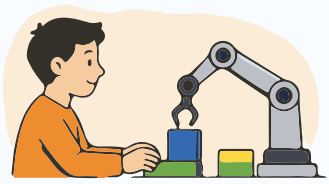
Robotul colectează informații din mediul înconjurător prin senzori.

2 DECIDE
program



Informațiile sunt procesate conform instrucțiunilor programului în vederea stabilirii acțiunii corespunzătoare.

3 ACȚIONEAZĂ
actuatoare



Robotul acționează asupra mediului prin actuatoare (motoare, brațe, roți, etc.)

SIMTE · DECIDE · ACȚIONEAZĂ

Cele trei funcții ale oricărui robot

1. **SIMTE.** Această funcție se numește percepție și este realizată prin senzori. Un senzor este o componentă electronică ce transformă o mărime fizică (intensitatea luminii, temperatura, distanța, accelerația) într-un semnal electric ce poate fi prelucrat de circuitele robotului. Exemplu: senzorul de distanță al unui aspirator robot măsoară distanța până la perete.

2. **DECIDE.** Informațiile primite de la senzori sunt prelucrate de un microcontroler: un circuit integrat care conține un procesor, memorie și porturi de intrare-ieșire. Microcontrolerul execută un program, adică o secvență finită de instrucțiuni scrisă de un om, pe care o parcurge în ordine, exact așa cum a fost formulată. Robotul nu improvizează niciodată; el execută doar instrucțiunile din program.

3. **ACȚIONEAZĂ.** Pe baza deciziei luate, robotul modifică starea mediului prin elemente de execuție (actuatoare): motoare, surse de lumină, difuzoare. Exemplu: motorul care rotește periile aspiratorului robot.

Testul celor trei întrebări

Pentru a stabili dacă un dispozitiv este robot, verifică dacă acesta îndeplinește simultan cele trei funcții: percepe? decide? acționează?

✓ **Mașina de spălat.** Măsoară greutatea rufelor printr-un senzor, selectează programul de spălare potrivit și acționează motorul care rotește tamburul. → Robot

✓ **Ușa automată.** Detectează apropierea unei persoane prin senzori de mișcare, decide să se deschidă și acționează motorul care deplasează ușa. → Robot.

✗ **Mixerul.** Acționează un motor, însă nu percepe nimic și nu ia nicio decizie. Prin urmare, nu este robot, ci o unealtă electrică.

Roboții din jurul nostru

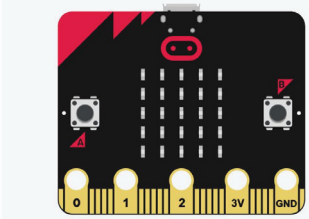
Imaginile de mai jos prezintă dispozitive uzuale care îndeplinesc toate cele trei funcții din definiția robotului, chiar dacă aspectul lor nu amintește de roboții din filme.

Fig. 1. Roboții au rareori aspect umanoid. Suntem înconjurați de ei în viața de zi cu zi.

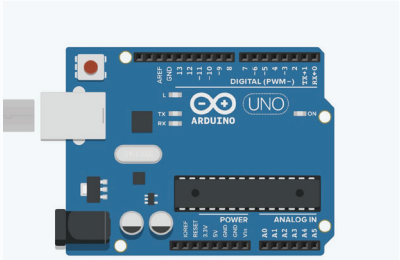
SECȚIUNEA 02 · PIESELE DE AZI

Cele două creiere ale manualului

Plăcuța Micro:bit



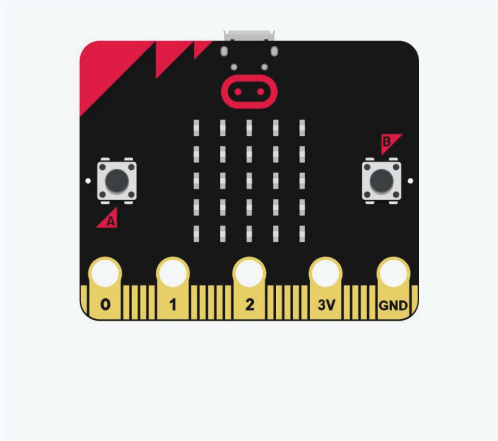
Piesă nouă: Plăcuța Arduino Uno PRIMA APARIȚIE



SECȚIUNEA 02 · PIESELE DE AZI

Plăcuța Micro:bit

MICRO:BIT



CE ESTE?

O placă de dezvoltare de dimensiuni reduse (aproximativ 5 cm × 4 cm), creată în Marea Britanie de BBC pentru educație. O placă de dezvoltare este un circuit electronic construit în jurul unui microcontroler, la care se pot conecta și programa cu ușurință alte componente.

CE FACE?

Execută programele scrise de utilizator și comandă componentele conectate la porturile sale. Include un afișaj format din 25 de LED-uri dispuse într-o matrice de 5×5, două butoane (A și B), un accelerometru, un magnetometru (busolă), un senzor de temperatură și unul de lumină. Tensiunea sa de lucru este de 3,3 V.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În școli din întreaga lume și în numeroase proiecte educaționale: alarme de cameră, stații meteo, mici console de jocuri.

DE CE NE TREBUIE AZI?

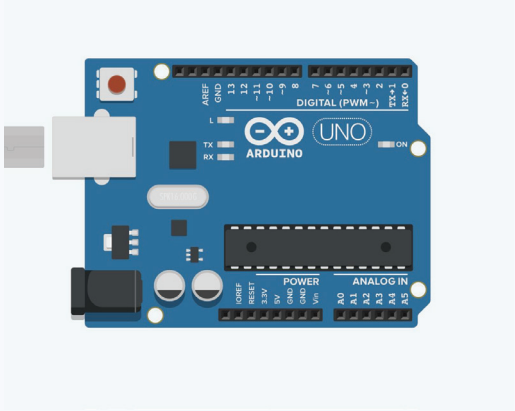
Este unul dintre cele două „creiere” utilizate în acest manual: componenta care realizează funcția de decizie a robotului. În Tinkercad, Micro:bit se programează în blocuri grafice sau în limbajul Python.

25 LED-uri · două butoane · senzori integrați · 3,3 V

SECȚIUNEA 02 · PIESELE DE AZI

Piesă nouă: Plăcuța Arduino Uno

PRIMA APARIȚIE



CE ESTE?

Cea mai răspândită placă de dezvoltare din lume. Proiectul Arduino a fost inițiat în Italia, în anul 2005, iar modelul Uno a fost lansat în anul 2010. Placa este construită în jurul microcontrolerului ATmega328P și reprezintă standardul atelierelor de electronică, al laboratoarelor universitare și al milioanelor de proiecte publicate pe internet.

CE FACE?

Execută programe și comandă componente, la fel ca Micro:bit. Nu are afișaj sau butoane încorporate, dar oferă 14 pini digitali și 6 intrări analogice, la o tensiune de lucru de 5 V. Un pin este un punct de conectare prin care placa transmite sau primește semnale electrice. Se programează în blocuri grafice sau în limbajul C++.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În imprimante 3D, în roboți de competiție, în instalații de artă interactivă: oriunde este necesar un dispozitiv electronic programabil.

DE CE NE TREBUIE AZI?

Este al doilea „creier” al manualului. Majoritatea proiectelor de electronică publicate pe internet sunt concepute pentru Arduino, astfel încât este util să îl cunoști încă de la început.

CE PLĂCUȚĂ UTILIZĂM ÎN GENERAL?

MICRO:BIT

Afișaj, butoane și senzori integrați direct pe placă; tensiune de lucru 3,3 V; se programează și în limbajul de programare Python, concepută pentru educație.

ARDUINO UNO

Fără afișaj încorporat, însă cu mai mulți pini și cu un curent disponibil mai mare pe fiecare pin; tensiune de lucru 5 V; limbajul său este C++, utilizat în ateliere și în industrie.

Concluzia: nu există „placa cea bună”, ci placa potrivită proiectului. Din acest motiv le studiezi pe amândouă și vei putea alege în cunoștință de cauză.

Și atelierul nostru: Tinkercad

Tinkercad este o platformă online gratuită, realizată de compania Autodesk, în care circuitele electronice se construiesc și se testează direct în browser. Componentele sunt virtuale, dar respectă comportamentul celor reale. Avantajele pentru un începător sunt considerabile: nicio componentă nu poate fi deteriorată, nu există risc de electrocutare, utilizarea este gratuită și este necesară doar o conexiune la internet. Testarea pe calculator a unui circuit, înainte ca acesta să fie construit fizic, se numește simulare. Inginerii utilizează simularea la fiecare proiect, deoarece aceasta permite descoperirea erorilor înainte de construcția fizică, reducând astfel costurile și riscurile.

SECȚIUNEA 03 · PRACTICĂ

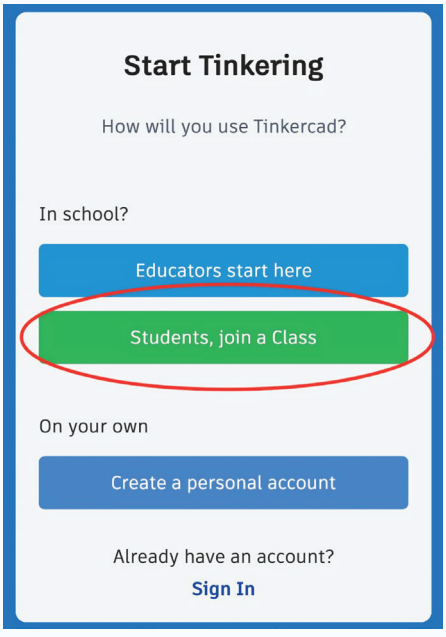
Prima ta simulare

CIRCUITUL DEMO AL LECȚIEI

▶ Deschide circuitul „Salut, robotică!”

După ce îți creezi contul (pasul 1 de mai jos), accesează linkul sau scanează codul QR, apoi apasă „Copy and Tinker” pentru a primi propria copie a circuitului. Pe planșa de lucru se află ambele plăci, fiecare cu un program pregătit în prealabil. Pe placa Micro:bit, programul comută între două moduri de afișare pe matricea de LED-uri, iar pe placa Arduino, programul aprinde și stinge LED-ul marcat cu „L”.

1 · Fă-ți cont pe Tinkercad



Intră pe tinkercad.com și apasă „Sign up” (înregistrare). Alege varianta destinată elevilor și folosește codul de clasă primit de la profesor sau o adresă de e-mail personală, dacă lucrezi de acasă. Procesul durează aproximativ două minute, iar contul este gratuit.

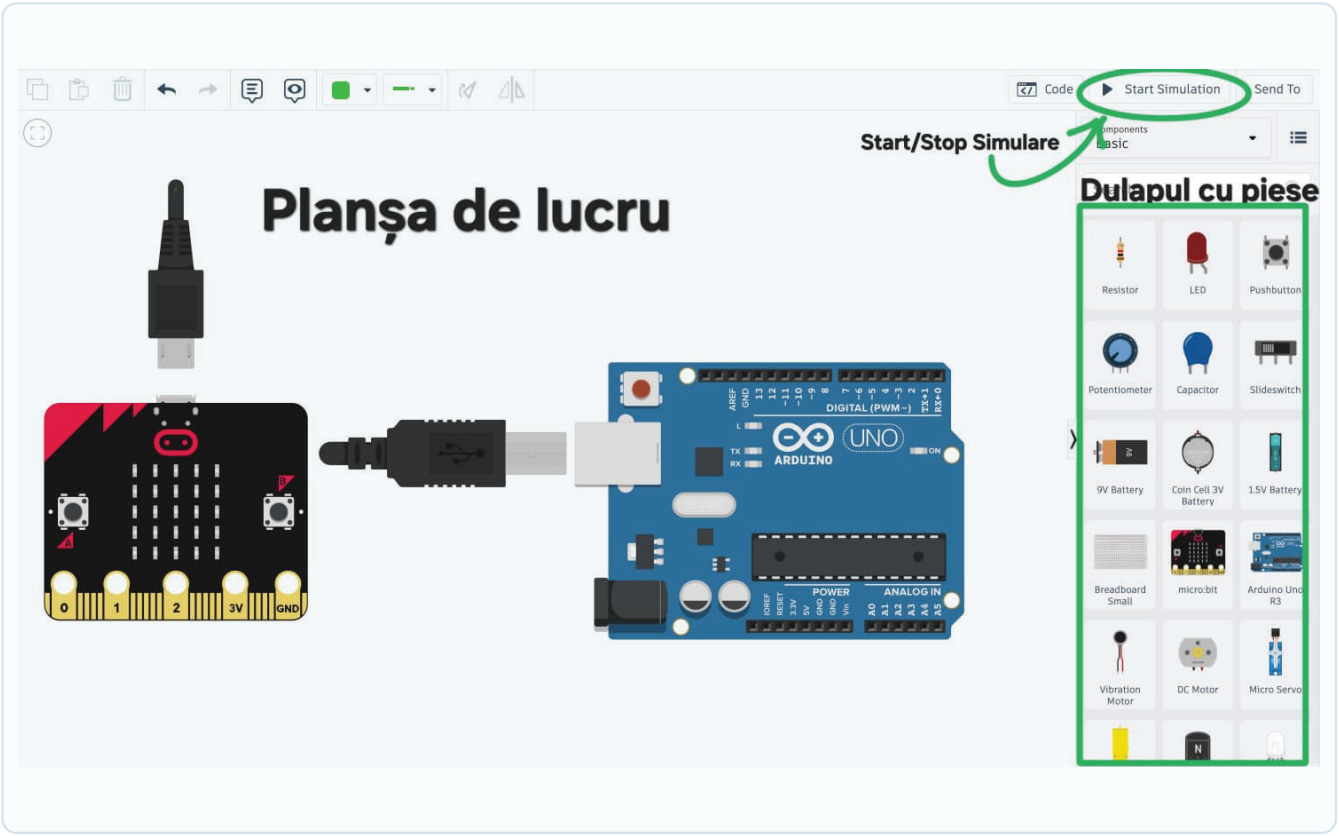
☐ Am făcut acest pas

SECȚIUNEA 03 · PRACTICĂ

Prima ta simulare

2 · Deschide circuitul demo și privește în jur

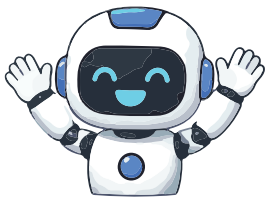
Folosește linkul sau codul QR de mai sus și apasă „Copy and Tinker”. Pe planșa de lucru din centru se află cele două plăci studiate anterior. În partea dreaptă se găsește dulapul cu piese (biblioteca de componente), iar în partea de sus, butoanele principale. Deocamdată doar observă interfața: în această lecție nu conectăm fire.



☐ Am făcut acest pas

3 · Apasă „Start Simulation”, apoi butonul A

Apasă butonul verde „Start Simulation” din colțul din dreapta sus. Ambele plăci pornesc: pe afișajul Micro:bit pulsează o inimă, iar pe Arduino clipește LED-ul marcat L de lângă procesor. Apasă apoi pe butonul A al plăcii Micro:bit din simulator: imaginea afișată se schimbă. Fiecare placă execută un program pregătit în prealabil. Peste cinci lecții, vei scrie chiar tu programul pe care îl va executa placa.



☐ Am văzut inima pulsând și luminița L clipind

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, toată clasa)

Pe ecran apar, pe rând, 10 dispozitive: mașină de spălat, mixer, dronă, aspirator robot, bicicletă, semafor cu senzori, uscător de păr, ușă automată de mall, ceas deșteptător clasic, automobil cu pilot automat. Pentru fiecare, clasa votează: robot sau nu? Verdictul se stabilește apoi împreună, prin aplicarea definiției: percepe? decide? acționează? Astfel, definiția robotului se consolidează prin exercițiu, nu prin dictare.

Varianta Kahoot: cele 10 dispozitive, sub formă de întrebări cu imagini; linkul și codul QR se află în folderul lecției.

Varianta pe foaie: fișa 1.1-J: un tabel cu cele 10 dispozitive și trei coloane de bifat: percepe / decide / acționează.

Varianta fără ecrane: profesorul numește dispozitivul, iar elevii ridică 1, 2 sau 3 degete, în funcție de câte dintre cele trei funcții are acesta; cine greșește își motivează alegerea, cine are dreptate o explică.

3 întrebări rapide: Notează mai jos ce ai descoperit

1. Care sunt cele trei funcții pe care le îndeplinește orice robot?
2. Mixerul din bucătărie este robot? Argumentează răspunsul.
3. Ce este o simulare și de ce o utilizează și inginerii profesioniști?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

robot: sistem tehnic programabil care percepe mediul, ia decizii conform unui program și acționează asupra mediului

senzor: componentă electronică ce transformă o mărime fizică (lumină, temperatură, distanță, accelerație) într-un semnal electric

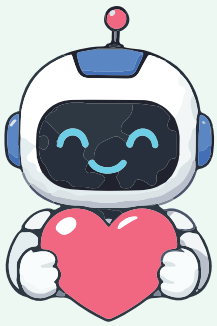
program: secvență finită de instrucțiuni pe care microcontrolerul le execută în ordine, exact cum au fost scrise

Micro:bit: placă de dezvoltare educațională cu afișaj de 25 de LED-uri, butoane și senzori integrați; se programează și în Python

Arduino Uno: placă de dezvoltare standard, construită în jurul microcontrolerului ATmega328P; se programează în C++

Tinkercad: platformă online gratuită pentru construirea și simularea circuitelor electronice

simulare: testarea pe calculator a comportamentului unui circuit, înainte de construcția fizică a acestuia



Ce trebuie să știi după această lecție

- ✓ Robot = percepe + decide + acționează; verifică orice dispozitiv aplicând definiția
- ✓ Robotul execută doar instrucțiunile din program, exact în ordinea scrisă
- ✓ Micro:bit și Arduino Uno sunt cele două plăci de dezvoltare ale manualului; Tinkercad este mediul de simulare în care lucrăm
- ✓ Simularea permite testarea fără riscuri și fără costuri; din acest motiv o utilizează și inginerii

LECȚIA 1.1

LECȚIA 1.2

Curentul electric și circuitul închis

În această lecție vei studia curentul electric, „sângele” oricărui robot, cu ajutorul unei analogii pe care o vom folosi în tot manualul și anume, analogia apei. Vei afla, de asemenea, de ce comutatorul din perete stinge lumina.

CE VEI ȘTI LA FINALUL LECȚIEI

- Să definești intensitate curentului electric și tensiunea electrică, folosind și analogia apei.
- Să enunți condiția fundamentală de funcționare: curentul circulă doar în circuit închis.
- Să explici, pe un circuit real, rolul întrerupătorului în aprinderea și stingerea becului.

SECȚIUNEA 01 · TEORIE

Curentul și tensiunea, pe înțelesul oricui

DEJA CUNOȘTI (DIN LECȚIA 1.1)

Robotul simte → decide → acționează. Toate cele trei funcții au însă nevoie de energie pentru a fi îndeplinite. Această energie este subiectul lecției de astăzi: energia electrică.

Curentul electric reprezintă deplasarea ordonată a purtătorilor de sarcină electrică printr-un conductor; care poate fi confecționat din diferite materiale conductoare, ca de exemplu metalul, acești purtători sunt electronii liberi. Mărimea fizică ce indică intensitatea acestei deplasări se numește **intensitatea curentului electric**, se notează cu I și se măsoară în amperi (simbol A).

Tensiunea electrică este mărimea fizică ce exprimă „presiunea” cu care sursa împinge purtătorii de sarcină prin circuit. Se notează cu U și se măsoară în volți (simbol V). Exemple de valori uzuale: o baterie de tip AA are 1,5 V, acumulatorul unui telefon are aproximativ 3,7 V, iar tensiunea rețelei electrice din România este de 230 V.

Analogia apei: o vom folosi în tot manualul

Un circuit electric poate fi comparat cu o instalație de apă: tensiunea (U) corespunde presiunii de la robinet, intensitatea curentului (I) corespunde debitului de apă, iar conductoarele (firele) corespund furtunului. Presiunea mare cu debit mic este o combinație posibilă, ca la un aparat de spălat cu jet subțire aflat sub presiune ridicată. De fiecare dată când în manual apare o componentă nouă, o vom traduce și în „limba apei”.

U presiune	I debit	= furtun

⚠ O singură regulă de siguranță, valabilă pe viață

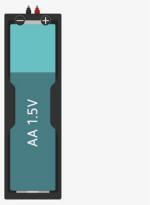
În acest manual lucrăm exclusiv cu tensiuni mici, considerate nepericuloase (sub 5 V) și, în plus, în simulator, unde niciun element nu poate provoca accidente. Rețeaua de 230 V nu este însă niciodată un loc de experimente: la această tensiune, curentul care poate străbate corpul uman devine periculos. Tot ceea ce ar fi riscant în realitate este permis și sigur în Tinkercad; tocmai de aceea îl folosim.

Regula de aur: cercul închis

Curentul electric poate circula doar dacă traseul său formează o buclă completă: pleacă de la sursă, străbate componentele și se întoarce la sursă. Ansamblul format din sursă, componente și conductoarele de legătură se numește **circuit electric** (termenul provine chiar de la „cerc”). Dacă traseul este întrerupt în orice punct, chiar și într-unul singur, curentul se oprește instantaneu în întregul circuit. Situația este asemănătoare unei piste de alergare rupte în două: nu se poate alerga „doar pe porțiunea bună”.

SECȚIUNEA 02 · PIESELE DE AZI

Piesă nouă: Bateria



CE ESTE?

O sursă de energie electrică ce transformă energia chimică înmagazinată în interiorul său în energie electrică. Din acest motiv, bateria se mai numește și generator electrochimic.

CE FACE?

Menține tensiunea care pune în mișcare purtătorii de sarcină; în analogia apei, este pompa instalației. Are două borne: prin convenție, sensul curentului în circuitul exterior este de la borna + (plus) către borna − (minus). Electronii se deplasează, în realitate, în sens opus.

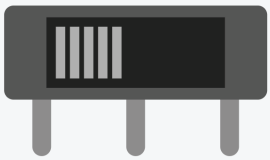
UNDE O ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În telecomandă, în ceasul de perete, în telefon, în automobil: orice dispozitiv care funcționează „fără fir” conține o baterie.

DE CE NE TREBUIE AZI?

Este sursa circuitului nostru demonstrativ: fără sursă, curentul nu poate circula, așa cum fără pompă apa nu curge.

Piesă nouă: Întrerupătorul



CE ESTE?

O componentă cu un contact mobil, amplasată pe traseul curentului, care unește sau separă două conductoare prin acționarea unei clapete sau a unui buton.

CE FACE?

Închide sau deschide circuitul printr-o singură mișcare. Contact stabilit = circuit închis = curentul circulă. Contact întrerupt = circuit deschis = curentul se oprește.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

Pe fiecare perete din locuință: comutatorul de lumină este exact acest dispozitiv, montat într-o carcasă.

DE CE NE TREBUIE AZI?

Pentru a întrerupe și a reface circuitul cu un simplu click, observând astfel regula circuitului închis în acțiune.

Piesă pe care o știi deja de-acasă: becul

Becul transformă energia electrică în energie luminoasă și termică. Îl folosim astăzi pentru că este o componentă simplă și nepolarizată: spre deosebire de componenta nouă din lecția următoare, becul funcționează indiferent de sensul în care este conectat în circuit.

SECȚIUNEA 03 · EXPLORARE

Rupe cercul cu mâna ta

CIRCUITUL DEMO AL LECȚIEI

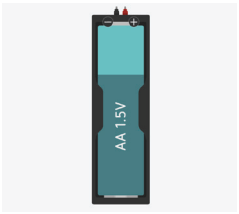
▶ Deschide circuitul „Cercul închis”

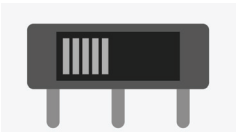
Circuitul este construit în prealabil: baterie → întrerupător → bec → înapoi la baterie. Apasă „Copy and Tinker” pentru a primi propria copie.

1 · Pornește simularea și acționează întrerupătorul

Apasă „Start Simulation”, apoi execută click pe clapeta întrerupătorului. Becul se aprinde, apoi se stinge. Ce se întâmplă, de fapt, la fiecare apăsare? Circuitul se închide, respectiv se deschide.

☐ **Am aprins și am stins becul**





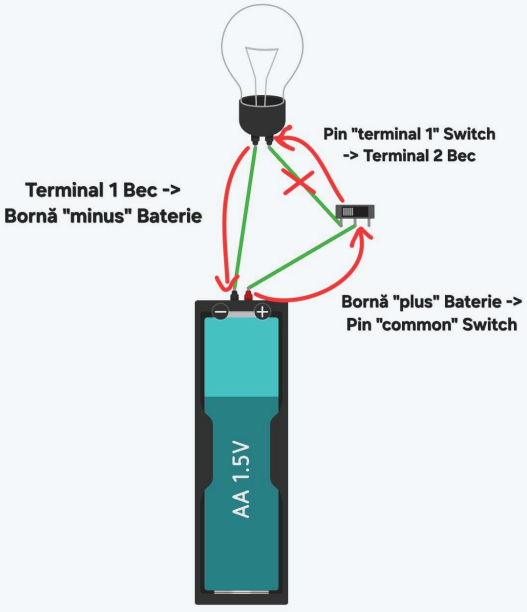
2 · Întrerupe circuitul: șterge un fir

Cu becul aprins, execută click pe oricare fir și apasă tasta Delete. Becul se stinge instantaneu, deși bateria este încărcată și becul este funcțional. Circuitul este întrerupt, prin urmare curentul s-a oprit peste tot, nu doar în porțiunea firului eliminat.

☐ **Am întrerupt circuitul**

3 · Repară circuitul

Apasă Ctrl+Z (anulează ultima acțiune) sau trasează un fir nou între cele două puncte rămase libere: click pe primul punct, apoi click pe al doilea. Lumina revine: circuitul este din nou complet.



☐ **Am reparat circuitul**

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, cu mișcare)

Clasa se așază în cerc, ținându-se de mâini: mâinile unite reprezintă firele, iar un elev ales este „becul” (când „curentul” ajunge la el, ridică brațele și strigă „APRINS!”). Profesorul este bateria: pornește o strângere de mână care se transmite din elev în elev, de-a lungul cercului.

Apoi profesorul strigă numele unui elev, iar acesta eliberează mâinile vecinilor: circuitul este întrerupt, „becul” coboară brațele și strigă „STINS!”, indiferent unde s-a produs întreruperea. Jocul se repetă cu întreruperi în locuri diferite, pentru a evidenția faptul că poziția întreruperii nu contează: efectul este întotdeauna total.

Varianta Kahoot: 6 scheme de circuite desenate, cu întrebarea „se aprinde sau nu?”; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 1.2-J: desenează un circuit care NU funcționează și cere-i colegului de bancă să identifice cauza

Varianta fără ecrane: chiar jocul principal; necesită doar participanți și spațiu între bănci

3 întrebări rapide

1. Definește tensiunea electrică și intensitatea curentului electric, folosind analogia apei.
2. De ce se stinge becul atunci când întrerupătorul deschide circuitul?
3. Care este, prin convenție, sensul curentului electric în circuitul exterior al unei baterii?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

curent electric: deplasarea ordonată a purtătorilor de sarcină electrică; intensitatea sa (I) se măsoară în amperi (A)

circuit: ansamblul format din sursă, componente și conductoarele de legătură, prin care circulă curentul

baterie: sursă care transformă energia chimică în energie electrică; „pompa” circuitului

întrerupător: componentă cu contact mobil care închide sau deschide circuitul

tensiune: mărimea fizică (U) ce exprimă „presiunea” cu care sursa împinge curentul; se măsoară în volți (V)

circuit închis / deschis: traseu complet (curentul circulă) / traseu întrerupt (curentul se oprește)

borne + și −: polii bateriei; prin convenție, curentul iese pe borna + și se întoarce pe borna −

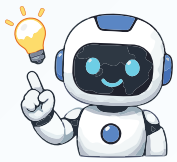
Ce trebuie să știi după această lecție:

- ✓ Tensiunea (U, în volți) împinge, curentul (I, în amperi) curge; reține analogia apei
- ✓ Curentul circulă doar în circuit închis; o întrerupere în orice punct oprește totul
- ✓ Bateria este sursa circuitului: prin convenție, curentul iese pe borna + și se întoarce pe borna −
- ✓ Întrerupătorul nu „taie curentul” în mod misterios: el doar deschide și închide circuitul

LECȚIA 1.3

LED-ul și rezistorul

În această lecție vei studia LED-ul, împreună cu partenerul său nedespărțit, rezistorul. Vei descoperi, de asemenea, pinii celor două plăci de dezvoltare.



CE VEI ȘTI LA FINALUL LECȚIEI

- Să definești LED-ul și rezistorul și să explici rolul fiecăruia în circuit.
- Să explici de ce LED-ul nu se conectează niciodată fără rezistor.
- Să identifici, pe fiecare placă, pinii comandați din program, pinii de alimentare (3V / 5V) și pinul GND.

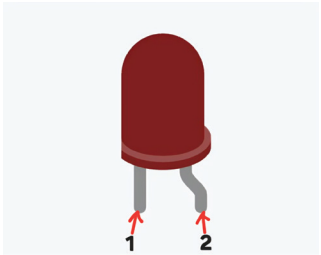
SECȚIUNEA 01 · PIESELE DE AZI

Componenta care luminează și componenta care o protejează

DEJA CUNOȘTI (DIN LECȚIA 1.2)

Curentul circulă doar în circuit închis, pus în mișcare de tensiune. Astăzi introducem în circuit două componente noi și vei afla de ce una dintre ele nu se conectează niciodată individual.

Piesă nouă: LED-ul



CE ESTE?

O componentă electronică semiconductoare, încapsulată, cu două terminale metalice, care emite lumină la trecerea curentului electric. Denumirea provine din limba engleză: Light-Emitting Diode, adică „diodă emițătoare de lumină”.

CE FACE?

Transformă energia electrică în energie luminoasă. Fiind o diodă, LED-ul este o componentă polarizată: permite trecerea curentului într-un singur sens. Pentru a deosebi terminalele, acestea au lungimi diferite: terminalul scurt (în imagine, piciorul „1”) este catodul, prin care curentul iese, iar terminalul lung (în imagine, piciorul „2”) este anodul, prin care curentul intră.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

Pretutindeni: indicatorul luminos al încărcătorului, luminile routerului, stopurile automobilelor, semafoarele moderne, pixelii ecranului pe care citești acum. Cele 25 de puncte luminoase de pe placa Micro:bit sunt tot LED-uri.

DE CE NE TREBUIE AZI?

Este cel mai simplu mod de a verifica vizual faptul că un circuit și, mai târziu, un program funcționează corect.

LED-ul nu își limitează singur curentul: conectat direct la sursă, prin el trece un curent mult mai mare decât cel maxim admis (de regulă, aproximativ 20 mA pentru LED-urile uzuale) și se arde în mai puțin de o secundă, printr-o supraîncălzire ireversibilă. De aceea, LED-ul nu se conectează niciodată singur: în același circuit cu el se află întotdeauna un rezistor, cu rol de protecție.

Piesă nouă: Rezistorul

CE ESTE?

O componentă electronică de forma unui mic cilindru, cu câte o bornă la fiecare capăt și inele colorate pe corp. Inelele formează codul culorilor, care indică valoarea rezistenței; în Tinkercad, codul nu este necesar: execuți click pe rezistor și introduci valoarea direct.

CE FACE?

Se opune trecerii curentului electric, limitându-i intensitatea; în analogia apei, are efectul unui deget apăsat pe furtun. Proprietatea sa se numește rezistență electrică, se notează cu R și se măsoară în ohmi, simbol Ω (litera grecească „omega”). Cu cât rezistența este mai mare, cu atât curentul care trece este mai mic.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În orice aparat electronic. Un telefon conține câteva sute de rezistori; fără ei, componentele s-ar arde una după alta.

DE CE NE TREBUIE AZI?

Ca protecție a LED-ului: limitează curentul la valoarea necesară unei iluminări corecte, fără a o depăși.

SECȚIUNEA 02 · TEORIE

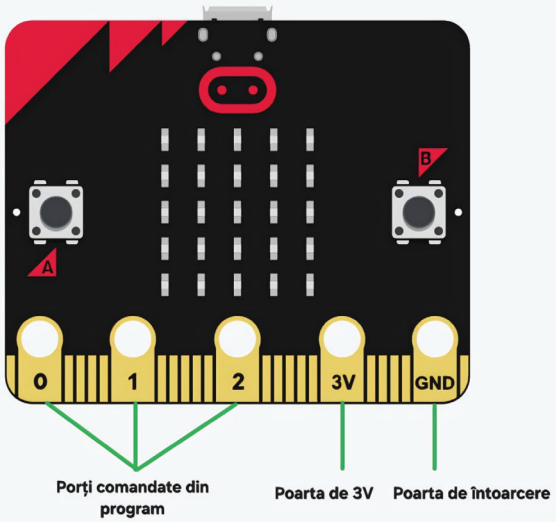
Pinii plăcilor de dezvoltare și GND

Avem componenta și protecția ei. Unde le conectăm însă? Pe marginea fiecărei plăci se află contacte metalice: fiecare se numește pin (termen englezesc care înseamnă „ac” sau „știft”). Pinii pot fi priviți ca niște porți numerotate prin care curentul poate ieși din placă sau se poate întoarce în ea. Trei roluri sunt importante acum și există pe ambele plăci:

ROLUL PINULUI	LA MICRO:BIT	LA ARDUINO UNO
Pini comandați din program; îi vom comanda din cod, începând cu: Lecția 1.6	Pinii P0, P1, P2	Pinii digitali 0–13 (cei marcați cu ~ au funcții suplimentare; le vom descoperi în Modulul 2)
Pinul de alimentare; furnizează permanent tensiunea plăcii („robinetul mereu deschis”)	3V (furnizează aproximativ 3,3 volți)	3,3V sau 5V (furnizează 3,3 sau 5 volți)
Pinul de întoarcere (masa); joacă rolul bornei − (minus) a bateriei	GND	GND (există mai mulți pini GND, toți echivalenți)

GND vine de la ground („pământ” în limba engleză); în electronică, acest punct comun de întoarcere se numește și masă. Orice curent care iese din placă trebuie să se întoarcă în ea prin GND, deoarece, după cum știi deja, curentul circulă doar în circuit închis. Placa îndeplinește acum și rolul de sursă, și pe cel de punct de întoarcere. Diferența dintre 3,3 V și 5 V nu este un detaliu: în lecția următoare vei vedea cum schimbă un calcul întreg.

LA MICRO:BIT



LA ARDUINO UNO

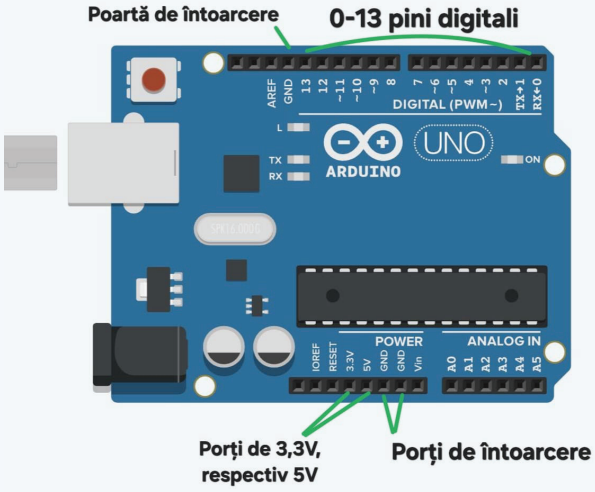


Fig. 1. Aceleași trei roluri, pe ambele plăci. În lecția următoare calculăm ce curent lăsăm să treacă prin pini, iar peste două lecții îi cablăm.

SECȚIUNEA 03 · EXPLORARE

Studiază componentele (virtual)

Astăzi nu cablăm nimic: doar facem cunoștință cu componentele. Deschide Tinkercad, apasă „Create new Circuit” (creează un circuit nou) și:

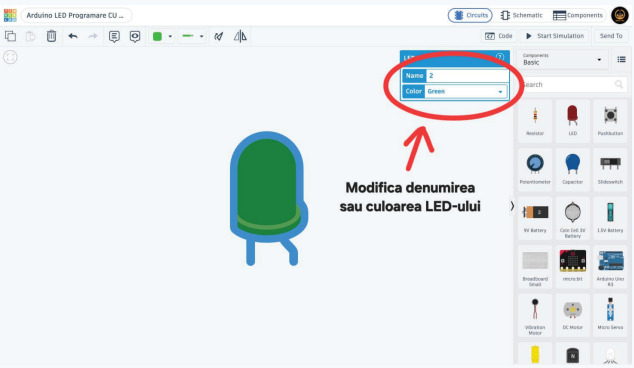
1 · Adu componentele pe planșă

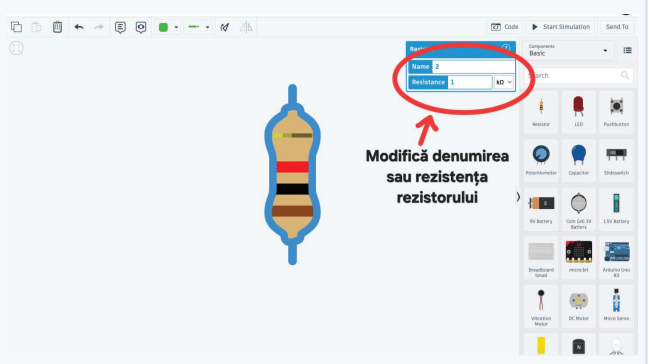
În dulapul cu piese din partea dreaptă, caută „LED” și trage-l pe planșa de lucru. Caută apoi „Resistor” (denumirea englezească a rezistorului) și așază-l alături. Recunoști la LED terminalele inegale prezentate în fișa componentei?

☐ **Ambele componente sunt pe planșă**

2 · Studiază-le de aproape

Rotește componentele cu tasta R, apoi apropie imaginea cu rotița mouse-ului. Execută click pe LED și schimbă-i culoarea din caseta de proprietăți; execută click pe rezistor și observă unde se introduce valoarea (o păstrăm pe cea propusă, deoarece valoarea corectă o vom calcula în lecția următoare).





☐ **Le-am rotit și le-am studiat**

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

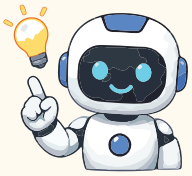
Cum se joacă (8 minute, pe echipe)

Profesorul citește ghicitori, iar echipele scriu răspunsul pe foaie și îl ridică la semnal. Exemple din folderul lecției: „Am două terminale inegale și funcționez doar dacă sunt conectat în sensul corect. Cine sunt?” (LED-ul) - „Sunt mic cât un bob de fasole, port inele și frânez tot ce trece prin mine.” (rezistorul) - „Numele meu înseamnă «pământ», dar stau pe o placă; prin mine totul se întoarce acasă.” (GND) · „Împing fără mâini, iar valoarea mea se măsoară în volți.” (tensiunea). Runda finală: echipele inventează propriile ghicitori și le adresează adversarilor.

Varianta Kahoot: ghicitorile și imagini cu componente de identificat; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 1.3-J, cu ghicitorile scrise și spațiu pentru ghicitoarea proprie

Varianta fără ecrane: jocul principal, doar cu voce și foi; poate fi organizat oriunde



Vânătoarea de LED-uri (temă-joc pentru acasă)

Până la ora următoare: numără câte LED-uri găsești în camera ta. Indiciu: caută indicatoarele mici de „standby”. La începutul Lecției 1.4 comparăm rezultatele; de obicei, câștigătorul depășește 20.

3 întrebări rapide

1. De ce nu conectăm niciodată LED-ul direct la sursă, fără rezistor?
2. Cum deosebești anodul de catodul unui LED?
3. Ce rol joacă pinul GND și ce tensiune furnizează pinul de alimentare al fiecărei plăci?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

LED: diodă emițătoare de lumină; permite trecerea curentului într-un singur sens

anod: terminalul lung al LED-ului; prin el intră curentul

rezistor: componentă care limitează curentul; rezistența sa (R) se măsoară în ohmi (Ω)

catod: terminalul scurt; prin el iese curentul, către GND

rezistor cu rol de protecție: rezistor conectat în serie cu o componentă sensibilă, pentru a o proteja de un curent prea mare

3V / 5V: pinul de alimentare, care furnizează permanent tensiunea plăcii

pin: contact metalic al plăcii, prin care curentul iese sau intră

GND: pinul de întoarcere („ground” = pământ), numit și masă; borna − a plăcii

Ce trebuie să știi după această lecție:

- ✓ LED-ul luminează doar conectat în sensul corect: terminal lung = anod = intrare, terminal scurt = catod = ieșire;
- ✓ Rezistorul limitează curentul; rezistența (R) se măsoară în ohmi (Ω);
- ✓ Conectat direct la sursă, LED-ul se arde aproape instantaneu; de aceea are întotdea una rezistor de protecție;
- ✓ Aceleași trei roluri de pini pe ambele plăci: comandați din program, alimentare 3V/5V, GND = întoarcerea.

LECȚIA 1.4

Legea lui Ohm: alegem rezistorul potrivit

De unde știm ce rezistor îi trebuie LED-ului? În această lecție vei învăța prima și cea mai importantă lege din electronică, pas cu pas, și o vei aplica pe ambele plăci.

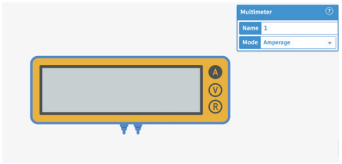
CE VEI ȘTI LA FINALUL LECȚIEI

- Să calculezi, cu ajutorul legii lui Ohm, care este rezistorul cu rol de protecție potrivit pentru orice LED;
- Să rotunjești rezultatul la o valoare standard, disponibilă la rezistorii din comerț;
- Să indici, fără calcul, valorile de siguranță pentru un LED conectat la micro:bit și la Arduino.

SECȚIUNEA 01 · PIESA DE AZI

Piesă nouă: Multimetrul

Piesă nouă: Multimetrul — PRIMA APARIȚIE



CE ESTE?

Aparatul de măsură universal al electronicii: un instrument cu afișaj și două sonde de măsurare, una roșie și una neagră. Denumirea provine de la faptul că reunește mai multe aparate într-unul singur: voltmetru, ampermetru și ohmmetru. În Tinkercad îl găsești în dulapul cu piese sub denumirea englezească „Multimeter”.

CE FACE?

Măsoară. Din caseta de proprietăți alegi mărimea dorită: tensiunea (V), intensitatea curentului (A) sau rezistența (Ω), iar valoarea apare pe afișaj. Pentru măsurarea curentului, aparatul se conectează în serie, adică direct pe traseul curentului: circuitul se întrerupe, iar aparatul se așază în locul întreruperii, astfel încât întregul curent să treacă prin el.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În trusa oricărui electrician și în orice atelier de robotică: cu el se depistează firul întrerupt, bateria descărcată sau siguranța arsă. Nu este o componentă a circuitului, ci instrumentul cu care îl observăm.

DE CE NE TREBUIE AZI?

Este singurul mod de a verifica dacă rezultatul calculului este corect. Calculul spune că prin circuit ar trebui să treacă 6 mA; multimetrul arată câți miliamperi trec în realitate.

SECȚIUNEA 02 · TEORIE

Legea care alege rezistorul

DEJA CUNOȘTI (DIN LECȚIILE 1.2 ȘI 1.3)

Tensiunea (U, în volți) împinge, curentul (I, în amperi) curge, rezistorul (R, în ohmi) limitează curentul. LED-ul fără rezistor de protecție se arde. Întrebarea de astăzi: ce valoare trebuie să aibă rezistența?

U=I×R

Legea lui Ohm

Răspunsul provine dintr-o lege formulată în anul 1827 de fizicianul german Georg Simon Ohm, numită în onoarea sa legea lui Ohm. Legea lui Ohm afirmă că intensitatea curentului printr-un conductor este direct proporțională cu tensiunea aplicată la capetele acestuia; matematic, U = I × R. Din această relație putem exprima rezistența: R = U ÷ I. Să aplicăm legea pe cazul nostru, pas cu pas:

Calculul rezistorului pentru LED, pe placa micro:bit

1. Ce știm despre placă: pinul de alimentare 3V furnizează o tensiune de aproximativ 3,3 V.

2. Ce știm despre LED: pentru a lumina, un LED roșu „reține” aproximativ 2 V din tensiunea disponibilă; această valoare se numește căderea de tensiune a LED-ului (tensiunea sa directă) și este indicată în fișa tehnică a componentei (o găsești și în Tinkercad).

3. Ce vrem: LED-ul luminează corect și are o durată mare de viață la un curent de aproximativ 0,006 A, adică 6 miliamperi (6 mA). Prefixul „mili” înseamnă a mia parte: 1000 mA = 1 A.

4. Gândim: pe rezistor trebuie să cadă tensiunea rămasă (3,3 − 2 = 1,3 V), la un curent de 0,006 A. Conform legii lui Ohm, rezistența necesară este egală cu tensiunea de pe rezistor, împărțită la curentul dorit.

R = (Uplacă − ULED) ÷ I → R = (3,3 V − 2,0 V) ÷ 0,006 A ≈ 217 Ω

Valorile standard din comerț (seria E12)

În comerț nu există rezistor de exact 217 Ω, la fel cum nu există pantofi cu mărimea 41,3. Producătorii fabrică valori fixe, organizate în serii standardizate (seria E12): ...100, 120, 150, 180, 220, 270, 330, 390, 470, 560, 680, 820, 1000 Ω... Regula practică: efectuează calculul, apoi alege prima valoare standard de deasupra rezultatului; o rezistență puțin mai mare înseamnă un curent puțin mai mic, deci mai multă siguranță. Pentru 217 Ω, prima valoare de deasupra este 220 Ω. Aceasta este valoarea aleasă.

...100, 120, 150, 180, 220, 270, 330, 390, 470, 560, 680, 820, 1000 Ω...

Aceeași lege, placa Arduino

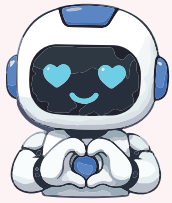
La Arduino, pinul de alimentare furnizează 5 V, nu 3,3 V. Nimic altceva nu se schimbă în raționament, doar prima valoare:

R = (5,0 V − 2,0 V) ÷ 0,006 A = 500 Ω → prima treaptă de deasupra: 560 Ω

Iată prima consecință practică a diferenței dintre 3,3 V și 5 V: aceeași lege, alt rezultat. O precizare onestă: pe internet vei întâlni frecvent LED-uri conectate la Arduino prin rezistoare de 220 Ω. Montajul funcționează, iar legea lui Ohm arată exact ce curent trece: 3 V ÷ 220 Ω ≈ 14 mA, valoare aflată încă sub limita de aproximativ 20 mA suportată de un LED obișnuit. LED-ul luminează mai puternic; noi păstrăm însă ținta de 6 mA, pentru a avea un singur mod de a raționa.

⚠ Respiră

Formula rămâne mereu în manual și nu trebuie memorată astăzi. Regulile practice de reținut: LED la Micro:bit → 220 Ω; LED la Arduino, la ținta noastră de 6 mA → 560 Ω. Orice valoare mai mare funcționează și ea, însă o rezistență mai mare înseamnă un curent mai mic, deci o lumină mai slabă.



Micro:bit

220 Ω

Arduino

560 Ω

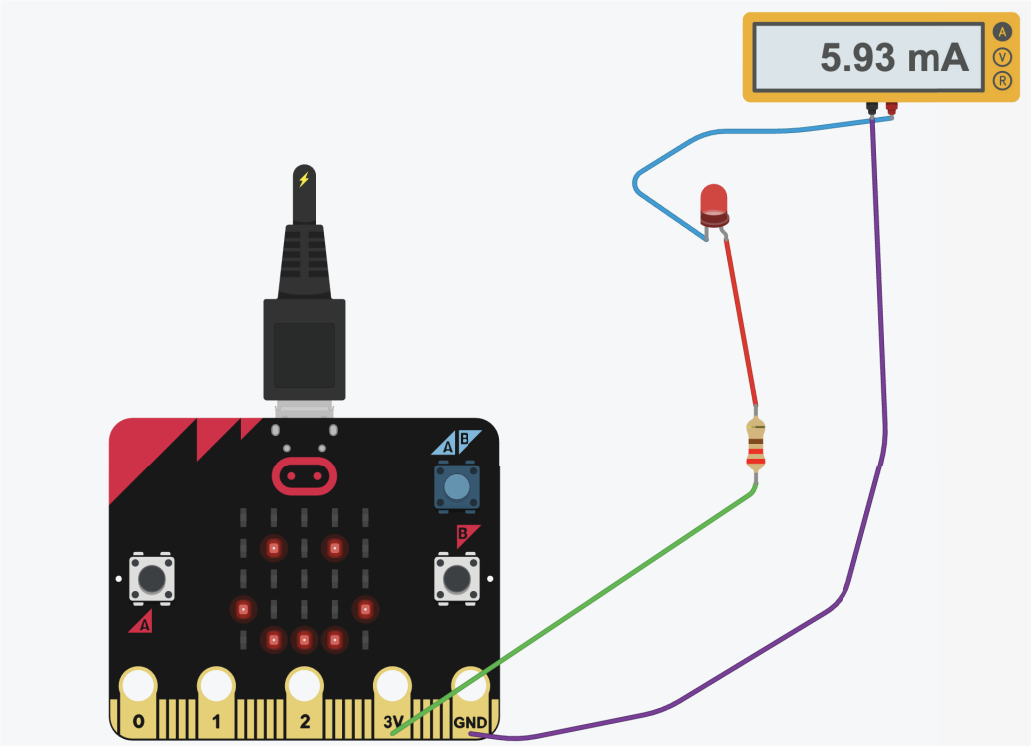
SECȚIUNEA 03 · VERIFICĂM ÎN TINKERCAD

Formula, verificată cu multimetrul

CIRCUITUL DEMO AL LECȚIEI

▶ Deschide circuitul „Cât curent trece, de fapt?”

Circuitul este construit în prealabil: pinul 3V al plăcii micro:bit → rezistor de 220 Ω → LED roșu → GND, cu un multimetru conectat în serie pentru măsurarea curentului. Apasă „Copy and Tinker” pentru a primi propria copie. Astăzi doar citești indicația aparatului; acest circuit îl vei construi singur, fir cu fir (fără multimetru, care nu este strict necesar), în Lecția 1.5.



1 **Pornește simularea și citește multimetrul**

Apasă „Start Simulation”. LED-ul se aprinde, iar multimetrul indică intensitatea curentului care trece efectiv prin circuit. Compară valoarea afișată cu cei 6 mA stabiliți în calcul.

☐ **Am citit valoarea de pe multimetru**

2 **Schimbă rezistorul și observă din nou**

Execută click pe rezistor și introdu 560 Ω în loc de 220 Ω. Pornește din nou simularea: intensitatea curentului scade, iar lumina slăbește. Rezistență mai mare = curent mai mic, exact conform legii lui Ohm.

☐ **Am încercat și cu 560 Ω**

SECȚIUNEA 04 · EXERSĂM

Rezolvă pe caiet, cu formula în față. Scrie calculul complet, nu doar rezultatul.

1 **Exercițiul 1 ·** LED verde la Micro:bit (3,3 V): cădere de tensiune 2,2 V, curent dorit 8 mA (0,008 A). Ce rezistor alegi?

2 **Exercițiul 2 ·** LED albastru la Micro:bit: cădere de tensiune 3,0 V, curent dorit 6 mA. Ce rezistor alegi? (Observă rezultatul cu atenție: de ce este atât de mic?)

3 **Exercițiul 3 ·** LED roșu la Arduino (5 V): cădere 2 V, curent dorit 10 mA. Ce rezistor alegi?

4 **Exercițiul 4 (invers) ·** Ai în sertar doar un rezistor de 330 Ω și un LED roșu (2 V), la micro:bit. Ce curent va trece? Se aprinde?

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, pe echipe)

Fiecare echipă primește 3 cartonașe-LED (pe fiecare sunt notate placa, culoarea, căderea de tensiune și curentul dorit) și are în față „piața”: cartonașe cu valorile standard (100, 150, 220, 330, 470, 560, 680 Ω...). Misiunea: calculează și „cumpără” rezistorul de protecție potrivit pentru fiecare LED. Prima echipă cu toate cele 3 cumpărături corecte câștigă. Punct bonus: echipa care poate explica de ce a rotunjit în sus, nu în jos.

Varianta Kahoot: calcule contra cronometru, cu 4 variante de rezistor la fiecare LED; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 1.4-J cu cartonașele de decupat (LED-uri + piața de rezistori)

Varianta fără ecrane: „piața vie”: 6 elevi devin rezistori (fiecare ține o foaie cu valoarea lui); echipele își strigă calculul și „cumpără” fizic rezistorul corect

Provocare: o întrebare de detectiv

Un coleg dorește să conecteze două LED-uri roșii unul după altul, adică în serie, la pinul 3V al plăcii Micro:bit. Nu se aprinde niciunul. De ce? Indiciu: adună căderile de tensiune ale celor două LED-uri și compară rezultatul cu tensiunea pe care o poate furniza placa.

Partea a doua: ar funcționa aceeași pereche de LED-uri la pinul 5V al plăcii Arduino? Efectuează calculul înainte de a răspunde; discutăm rezolvarea la începutul orei viitoare.

3 întrebări rapide

1. Scrie formula cu care alegi rezistorul de protecție al unui LED.
2. De ce rotunjim rezultatul mereu în sus, la prima valoare standard?
3. Care sunt valorile de siguranță de reținut pentru Micro:bit și pentru Arduino?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

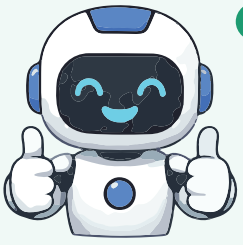
legea lui Ohm: relația U = I × R dintre tensiune, intensitatea curentului și rezistență

cădere de tensiune: tensiunea care rămâne pe o componentă atunci când aceasta este parcursă de curent

miliamper (mA): a mia parte dintr-un amper; 1000 mA = 1 A

valoare standard: valorile fixe în care se fabrică rezistorii (seria E12); rotunjim întotdeauna în sus

Ce trebuie să știi după această lecție:



- ✓ R = (Uplacă − ULED) ÷ I;
- ✓ Rotunjim întotdeauna în sus, la prima valoare standard: rezistență mai mare = siguranță mai mare;
- ✓ O rezistență prea mare nu distruge nimic: doar face lumina mai slabă;
- ✓ Regulile rapide: LED la micro:bit → 220 Ω; LED la Arduino (ținta de 6 mA) → 560 Ω.

LECȚIA 1.5

Primul circuit adevărat

În această lecție se leagă totul: LED-ul, rezistorul cu rol de protecție, pinii plăcii și regula circuitului închis. Vei construi primul tău circuit complet și îl vei vedea luminând.

CE VEI ȘTI LA FINALUL LECȚIEI

- Să construiești singur, în Tinkercad, circuitul: alimentare → rezistor → LED → GND;
- Să conectezi LED-ul în sensul corect;
- Să depanezi metodic un circuit care nu funcționează.

SECȚIUNEA 01 · PRACTICĂ

Construim, pas cu pas

DEJA CUNOȘTI (DIN LECȚIILE 1.3 ȘI 1.4)

Terminalul lung al LED-ului (anodul) primește curentul, iar cel scurt (catodul) îl trimite spre GND. Rezistorul de protecție calculat: 220 Ω la micro:bit, 560 Ω la Arduino. Curentul are nevoie de un circuit închis: astăzi, traseul va fi alimentare → rezistor → LED → GND.

Schema, comună ambelor trasee: pinul de alimentare (3V / 5V) → rezistorul de protecție → anodul LED-ului; catodul → GND. În Tinkercad, firele se conectează direct la pinii plăcii; un fir se trasează cu două clickuri: unul pe punctul de plecare, unul pe destinație.

TRASEUL A · MICRO:BIT

Alimentarea: pinul 3V · rezistorul de protecție: 220 Ω

CIRCUIT DE START

▶ Deschide circuitul de start 1.5 · Micro:bit

Placa Micro:bit se află deja pe planșa de lucru. Apasă „Copy and Tinker” și construiește în propria copie.

1 · Adu LED-ul pe planșă

Caută „LED” în dulapul cu piese și trage-l lângă placă. Verifică vizual terminalele: care este anodul? (Cel lung; el va primi curentul.)

☐ **LED-ul este pe planșă**

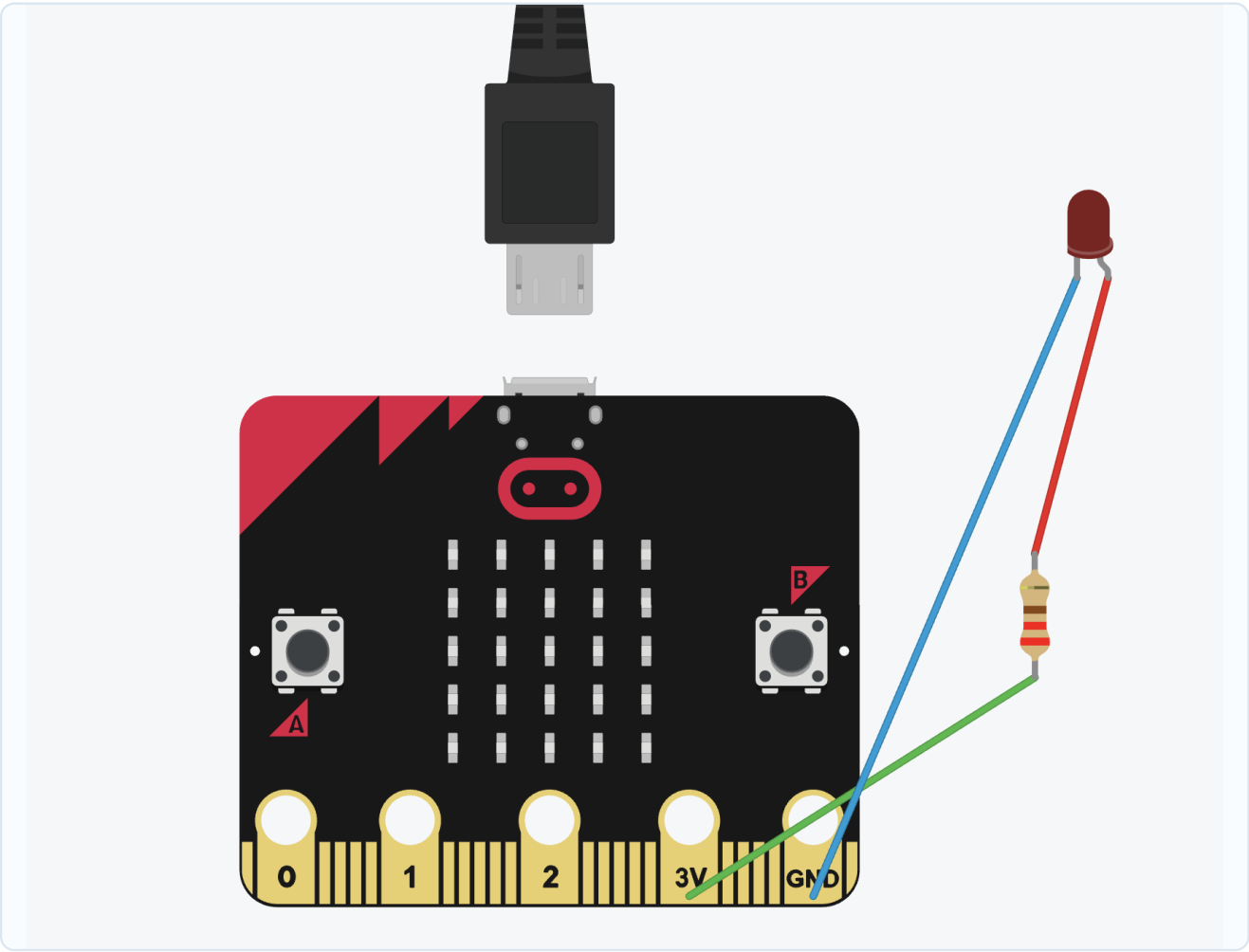
2 · Adu rezistorul și setează-l la 220 Ω

Caută „Resistor”, trage-l pe planșă, execută click pe el și introdu în caseta de proprietăți valoarea calculată în lecția trecută: 220. Atenție la unitate: să fie Ω, nu kΩ! (1 kΩ = 1000 Ω; o rezistență de aproximativ 4,5 ori mai mare ar face LED-ul aproape invizibil.)

☐ **220 Ω setat corect**

3 · Trasează cele 3 fire: închide circuitul

Firul 1: pinul 3V a plăcii → un capăt al rezistorului. Firul 2: celălalt capăt al rezistorului → terminalul lung (anodul) LED-ului. Firul 3: terminalul scurt (catodul) → pinul GND. Circuitul este închis!



☐ **Cele 3 fire sunt la locul lor**

TRASEUL B · ARDUINO UNO — **Alimentarea: pinul 5V · rezistorul de protecție: 560 Ω**

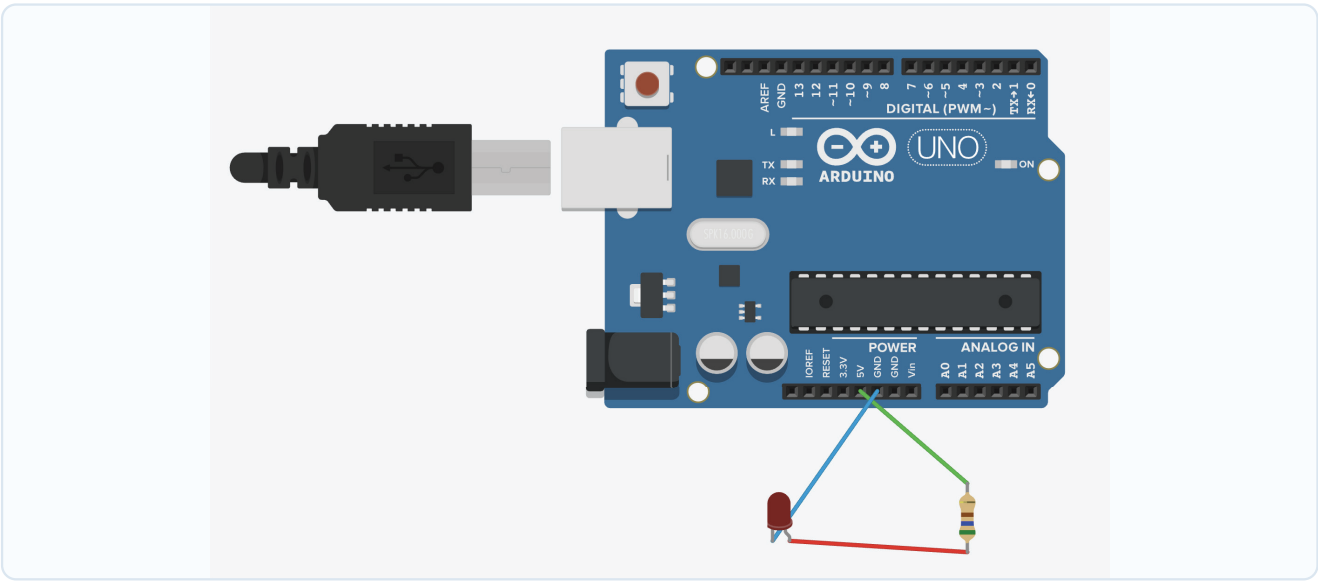
CIRCUIT DE START

▶ Deschide circuitul de start 1.5 · Arduino

Placa Arduino Uno se află deja pe planșa de lucru. Apasă „Copy and Tinker” și construiește în propria copie.

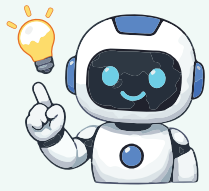
1 · Aceeași construcție, cu valorile plăcii tale

Adu LED-ul și rezistorul, setează rezistorul la 560 Ω (valoarea calculată pentru 5 V), apoi trasează cele 3 fire: pinul 5V → rezistor → anod; catod → GND. Observă ce se schimbă față de traseul A: doar pinul de alimentare și valoarea rezistorului.



☐ **Cele 3 fire sunt la locul lor**

Apasă „Start Simulation”. LED-ul se aprinde și rămâne aprins: pinul de alimentare 3V / 5V furnizează tensiune permanent, după cum știi deja. Privește-l o clipă: este prima lumină aprinsă de un circuit construit de tine. În lecția următoare vei învăța LED-ul să clipească.



CARE ESTE PLĂCUȚĂ POTRIVITĂ AICI?

MICRO:BIT

Alimentare de 3,3 V → rezistor de 220 Ω. Tensiune mai mică, rezistență mai mică.

ARDUINO UNO

Alimentare de 5 V → rezistor de 560 Ω. Tensiune mai mare, rezistență mai mare.

Concluzia: aceeași construcție, alt calcul. Prima dovadă practică a Lecției 1.4: valoarea rezistorului nu este un număr magic, ci rezultatul unei legi care ține cont de placa folosită.

⚠ Greșeala clasică a acestei lecții: LED-ul conectat invers

Terminalul lung (anodul) trebuie orientat spre rezistor, iar cel scurt (catodul) spre GND. Conectat invers, nimic nu se deteriorează: LED-ul pur și simplu nu se aprinde, deoarece permite trecerea curentului într-un singur sens.

SECȚIUNEA 02 · DEPANARE

Nu funcționează? Verifică următoarele aspecte, în ordine

⚠ LED-ul nu se aprinde deloc

1) Sensul LED-ului: terminalul lung spre rezistor, cel scurt spre GND?

2) Firul pleacă exact din pinul 3V (respectiv 5V), nu din alt pin?

3) Circuitul este complet: toate cele 3 fire sunt conectate în punctele lor?

4) Ai apăsat „Start Simulation”? (Da, ni se întâmplă tuturor.)

⚠ LED-ul luminează foarte, foarte slab

Verifică unitatea de măsură a rezistorului: este setată pe kΩ în loc de Ω? 220 kΩ înseamnă o rezistență de 1000 de ori mai mare decât cea calculată. Schimbă unitatea din caseta de proprietăți. Ești blocat complet? Deschide circuitul-checkpoint, construit în prealabil, și compară-l fir cu fir cu al tău: ▶ Circuit checkpoint 1.5 (traseul tău) [+ QR în varianta finală].

SECȚIUNEA 03 · JOCULEȚUL LECȚIEI

Cum se joacă (6 minute, pe echipe)

Profesorul afișează pe rând 4 capturi de ecran cu circuite care NU funcționează (în folderul lecției): LED conectat invers · fir plecat din pinul 0 în loc de 3V · circuit neînchis · rezistor setat pe 220 kΩ în loc de 220 Ω. Echipele au 30 de secunde pe imagine pentru a scrie diagnosticul. 1 punct pentru fiecare greșeală găsită + 1 punct bonus pentru explicația „de ce nu funcționează”.

Varianta Kahoot:

aceleași 4 imagini, cu câte 4 variante de răspuns; linkul și codul QR se află în folderul lecției

Varianta pe foaie:

fișa 1.5-J, cu circuitele tipărite și spațiu de diagnostic

Varianta fără ecrane:

„doctorul de circuite”: un elev descrie verbal un circuit greșit („am conectat terminalul scurt la 3V...”), iar ceilalți strigă diagnosticul

Provocare: fără pași. Doar tu și circuitul.

Nivel 1 · Al doilea LED.

Adaugă un al doilea LED, aprins și el continuu, cu propriul său rezistor de protecție.

(Fiecare LED are rezistorul lui: protecțiile nu se împart.)

Nivel 2 · Schimbă culoarea.

Execută clic pe LED și schimbă-i culoarea în albastru.Luminează mai slab: de ce?

Indiciu: recitește Exercițiul 2 din Lecția 1.4 și compară căderile de tensiune.

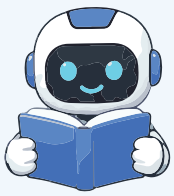
Nivel 3 · Oaspete din Lecția 1.2.

Adaugă în circuit un întrerupător (caută „Slideswitch”), pentru a aprinde și a stinge LED-ul manual, fără a opri simularea. Unde îl amplasezi?

(Surpriză: oriunde pe traseu, efectul este același. Poți explica de ce?)

3 întrebări rapide

1. Descrie, din memorie, traseul complet al circuitului tău, de la alimentare până la GND.
2. După ce recunoști un LED conectat invers și ce se întâmplă (sau nu) cu el?
3. LED-ul abia se vede: care este prima ta suspiciune și unde o verifici?



Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

Copy and Tinker: butonul care creează propria copie a unui circuit partajat

planșa de lucru: zona de lucru din Tinkercad în care așezi și conectezi componentele

cablare: conectarea componentelor prin fire, pentru a închide circuitul

checkpoint: circuit „de salvare”, construit în prealabil, de la care poți continua dacă te blochezi

kΩ (kiloohm): 1000 de ohmi; atenție la unitatea de măsură când setezi rezistorul

Ce trebuie să știi după această lecție

- ✓ Circuitul complet: 3V/5V → rezistor de protecție → anod → catod → GND;
- ✓ Sensul LED-ului decide totul: terminalul lung primește curentul, cel scurt îl trimite spre GND;
- ✓ Depanarea se face metodic: sens → pin corect → circuit complet → simulare pornită;
- ✓ Fiecare LED are propriul rezistor de protecție; acesta nu se împarte între două LED-uri.

LECȚIA 1.6

Primul tău program: LED-ul care clipește

Un singur fir mutat, iar circuitul tău trece de la „mereu aprins” la „ascultă de program”. În această lecție vei scrie primul tău program, în blocuri și în Python, și vei încheia Modulul 1.

CE VEI ȘTI LA FINALUL LECȚIEI

- Să explici diferența dintre pinul de alimentare (3V / 5V) și un pin comandat din program;
- Să scrii un program care aprinde și stinge LED-ul, pe traseul tău;
- Să modifici ritmul clipirii schimbând o singură valoare.

SECȚIUNEA 01 · JOC DE ÎNCĂLZIRE

Cum se joacă (8 minute, toată clasa)

Un elev voluntar devine „procesorul”: iese în fața clasei și execută literal, în ordine, exact instrucțiunile scrise de colegi pe tablă, de exemplu: „ridică mâna stângă” → „așteaptă 2 secunde” → „coboară mâna” → „repetă de 3 ori”. Partea amuzantă apare la instrucțiunile imprecise: „du-te la ușă” fără „oprește-te” înseamnă mers în ușă la nesfârșit! Concluzia pe care clasa o trăiește direct: robotul execută doar ce este scris, exact în ordinea scrisă; aceasta este cea mai importantă lecție de programare, predată fără niciun calculator. Cu mintea pregătită astfel, primul program de mai jos devine simplu.

Varianta Kahoot: „ce face acest robot?”: miniprograme de 3-4 instrucțiuni de interpretat; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 1.6-J: scrie instrucțiunile pentru „robotul care își pune ghiozdanul în bancă”; colegul le execută pe hârtie

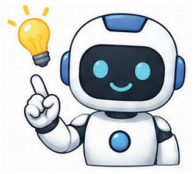
Varianta fără ecrane: chiar jocul principal; necesită doar tablă, cretă și un voluntar curajos

SECȚIUNEA 02 · TEORIE SCURTĂ

Un fir mutat schimbă tot

DEJA CUNOȘTI (DIN LECȚIILE 1.1, 1.3 ȘI 1.5)

Un program = o secvență de instrucțiuni executate în ordine, exact cum au fost scrise (ai văzut acest lucru în direct acum 5 minute!). Pinul 3V / 5V furnizează curent permanent; pinii comandați ascultă de program. Circuitul tău din Lecția 1.5 te așteaptă în contul tău Tinkercad.



Un fir mutat schimbă tot

Circuitul din lecția trecută rămâne aprins permanent deoarece este conectat la pinul de alimentare. Pentru a-l putea comanda, mutăm firul la un pin comandat din program: la Micro:bit, pinul 0; la Arduino, pinul 13 (bonus: LED-ul L de pe placă este conectat intern tot la pinul 13, deci va clipi împreună cu LED-ul tău). Din program îi vom transmite pinului „permite trecerea curentului” (scriem valoarea 1, numită și HIGH) sau „oprește curentul” (scriem valoarea 0, numită și LOW). Clipirea este doar atât, repetat: pornit → pauză → oprit → pauză.

HIGH / 1 → **pauză** → **LOW / 0** → **pauză**

TRASEUL A · MICRO:BIT — **Mutarea: firul din 3V → pinul 0 · Blocuri, apoi Python**

1 Singura modificare la circuit

Deschide circuitul tău din Lecția 1.5 (sau checkpoint-ul de mai jos). Prinde capătul firului conectat la 3V și mută-l în pinul 0. Atât. Dacă pornești simularea acum, LED-ul rămâne stins: pinul 0 așteaptă instrucțiuni. Să i le dăm.

☐ **Firul este mutat în pinul 0**

CHECKPOINT (DACĂ NU MAI AI CIRCUITUL DIN 1.5)

▶ Circuit gata cablat, cu firul deja mutat în pinul 0

Apasă „Copy and Tinker” și treci direct la cod.

2 Construiește programul în Blocuri

Apasă butonul „Code” din colțul din dreapta sus. Se deschide zona de programare, cu două moduri de lucru; le încercăm pe amândouă, în această ordine. Întâi Blocuri (piese colorate care se îmbină): forever → digital write pin P0 to HIGH → wait 500 ms → digital write pin P0 to LOW → wait 500 ms. Blocul forever („pentru totdeauna”) repetă la nesfârșit tot ce este așezat în el; wait 500 ms înseamnă „așteaptă 500 de milisecunde”. O milisecundă este a mia parte dintr-o secundă, deci

☐ **Clipește din blocuri**

3 Tradu-l în Python

Comută pe Python din meniul derulant al zonei de cod și scrie același program ca text; îl găsești mai jos, explicat rând cu rând. Indentarea, adică cele 4 spații cu care sunt împinse spre dreapta rândurile din interiorul blocului repetat, îi arată limbajului Python care rânduri aparțin acelui bloc.

☐ **Clipește și din Python**

ACELEAȘI ORDINE, CA TEXT — **PYTHON · MICRO:BIT**

```
# Lecția 1.6: LED care clipește
# (varianta Tinkercad / MakeCode)
def on_forever():   # definim un „bloc de instrucțiuni” cu numele on_forever
    pins.digital_write_pin(DigitalPin.P0, 1)   # deschide pinul P0 → LED aprins
    basic.pause(500)   # așteaptă 500 ms = jumătate de secundă
    pins.digital_write_pin(DigitalPin.P0, 0)   # închide pinul P0 → LED stins
    basic.pause(500)   # pauză și pe stins
basic.forever(on_forever)   # spune plăcii: repetă la nesfârșit blocul de mai sus
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

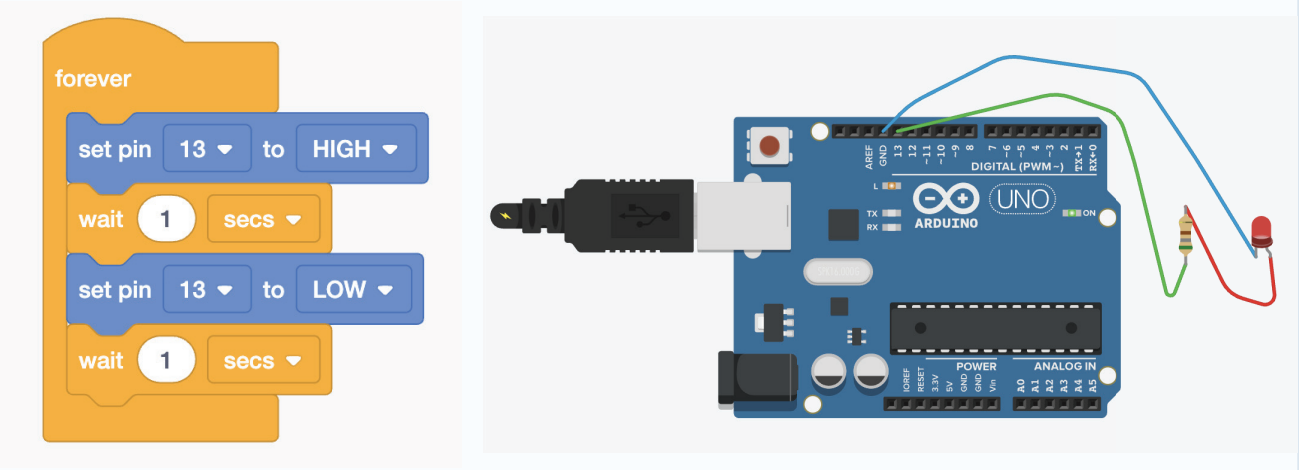
Copierea din PDF poate deteriora spațiile de la începutul rândurilor (indentarea), iar Python este strict în privința lor. Folosește linkul dacă preferi să lipești codul în loc să îl scrii.

TRASEUL B · ARDUINO UNO

Mutarea: firul din 5V → pinul 13 · Blocuri + C++

1 Mută firul și construiește blocurile

Deschide circuitul tău Arduino din Lecția 1.5 (sau checkpoint-ul). Mută firul din 5V în pinul 13. Apasă „Code” și alege modul „Blocks + Text”. Construiește: forever → set pin 13 to HIGH → wait 500 ms → set pin 13 to LOW → wait 500 ms. HIGH și LOW sunt denumirile valorilor logice 1 și 0 în mediul Arduino. Rulează: LED-ul tău și LED-ul L de pe placă clipesc împreună.



☐ **Clipesc amândouă**

CHECKPOINT (DACĂ NU MAI AI CIRCUITUL DIN 1.5)

▶ Circuit Arduino gata cablat, cu firul deja mutat în pinul 13

Apasă „Copy and Tinker” și treci direct la cod.

CE A GENERAT TINKERCAD DIN BLOCURILE TALE — **C++ · ARDUINO**

```
// C++ code
// Lecția 1.6: LED care clipește (varianta Arduino)
void setup() {   // se execută o singură dată, la pornire
  pinMode(13, OUTPUT);   // pinul 13 devine pin de ieșire
}
void loop() {   // echivalentul lui while True: se repetă la nesfârșit
  digitalWrite(13, HIGH);   // deschide pinul → LED aprins
  delay(500);   // delay = sleep: pauză în milisecunde (500 ms = 0,5 secunde)
  digitalWrite(13, LOW);   // închide pinul → LED stins
  delay(500);   // pauză și pe stins
}
```

COD GATA DE COPIAT

▶ Deschide codul C++

Folosește linkul dacă preferi să lipești în loc să scrii.

Dacă LED-ul clipește, oprește-te o clipă: tocmai ai scris primul tău program care controlează un circuit electric. Exact așa, doar cu mai multe rânduri de cod, funcționează semaforele, mașinile de spălat și roboții de pe Marte.

SECȚIUNEA 03 · DEPANARE

⚠ Nu clipește deloc?

1) Firul este conectat exact la pinul 0 (respectiv pinul 13)? Dacă a rămas la 3V / 5V, LED-ul rămâne aprins continuu și ignoră programul.

2) Ești în modul de cod corect și programul este cel de mai sus?

3) Simularea este pornită?

⚠ Python dă eroare?

În 9 din 10 cazuri este vorba despre indentare: cele 4 rânduri din interiorul blocului repetat (în codul nostru, rândurile de sub „def on_forever():”) trebuie împinse cu exact 4 spații, toate la fel. Verifică și cele două puncte (:) de la finalul rândului care deschide blocul.

SECȚIUNEA 04 · PROVOCARE

Fără pași. Doar tu și codul.

Nivel 1 · Semnalizatorul. Fă LED-ul să clipească exact ca semnalizarea unei mașini: aprins scurt (300 ms), stins mai lung (700 ms). Ce valori schimbi?

Nivel 2 · Clipirea alternativă. Adaugă al doilea LED (cu propriul rezistor de protecție) la pinul 1 (respectiv pinul 12) și fă-le să clipească alternativ: când unul este aprins, celălalt este stins. Indiciu: în codul Python al lecției, pinul 1 se numește DigitalPin.P1.

Nivel 3 · SOS. Semnalul SOS din codul Morse: 3 clipiri scurte, 3 lungi, 3 scurte, pauză mare, apoi de la capăt. Stabilește mai întâi duratele pe hârtie, apoi scrie codul.

3 întrebări rapide

1. De ce a trebuit mutat firul din 3V / 5V într-un pin comandat din program?
2. Ce rol are construcția „repetă la nesfârșit” și cum se numește ea în blocuri, în Python și în C++?
3. Ce se întâmplă dacă schimbi ambele valori “500” în “100”?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

Blocuri: programare vizuală cu piese colorate care se îmbină; ideală pentru primul program

Python: limbajul de programare text pe care îl folosim la Micro:bit, în tot manualul

C++: limbajul plăcii Arduino; Tinkercad îl generează automat din blocuri

while True: / forever / loop: trei denumiri pentru aceeași idee: „repetă la nesfârșit”

indentare: spațiile de la începutul rândului; indică limbajului Python ce rânduri formează un bloc

write_digital / digitalWrite: scrie 1 (HIGH) pe un pin pentru a-l activa și a aplica tensiune componentei conectate; scrie 0 (LOW) pentru a-l dezactiva și a opri alimentarea acelei componente

sleep(...) / delay(...): pune programul în pauză; valoarea se exprimă în milisecunde

milisecundă (ms): a mia parte dintr-o secundă; 500 ms = jumătate de secundă

Ce trebuie să știi după această lecție:

- ✓ Pinul 3V / 5V furnizează permanent; pinii comandați (0 la micro:bit, 13 la Arduino astăzi) ascultă de program;
- ✓ Blocurile, Python și C++ transmit aceleași instrucțiuni; se schimbă doar forma;
- ✓ Clipire: aprinderea și stingerea repetată a LED-ului, cu pauze între cele două stări;
- ✓ Indentarea nu este un decor: ea indică limbajului Python ce anume se repetă.

Ai terminat Modulul 1!

Ai început de la zero, iar astăzi ai construit un circuit pe două plăci diferite și l-ai programat în blocuri, în Python și în C++. În Modulul 2, curentul începe să pună lucrurile în mișcare: motoare, reductoare, servomotoare, până la primul mecanism complet dezvoltat de tine.

MODULUL 1 · ANEXA ELEVULUI

Răspunsuri la „3 întrebări rapide”

Lecția 1.1

1. 1. Percepe (prin senzori), decide (conform programului) și acționează (modifică lumea reală prin elemente de execuție).
2. 2. Nu: acționează, dar nu percepe și nu decide nimic; este doar o unealtă electrică.
3. 3. Testarea pe calculator a comportamentului unui circuit, înainte de construcția fizică; inginerii o folosesc pentru a descoperi erorile fără riscuri și fără costuri.

Lecția 1.2

1. 1. Tensiunea (U, în volți) este „presiunea” care împinge purtătorii de sarcină, precum presiunea de la robinet; intensitatea curentului (I, în amperi) corespunde debitului de apă.
2. 2. Deoarece circuitul se deschide: curentul se oprește instantaneu în întregul circuit, nu doar la întrerupător.
3. 3. Prin convenție, de la borna + către borna −, prin circuitul exterior (electronii circulă în sens opus).

Lecția 1.3

1. 1. Deoarece LED-ul nu își limitează singur curentul: prin el ar trece un curent mult peste cel maxim admis (aproximativ 20 mA) și s-ar arde în mai puțin de o secundă.
2. 2. Anodul este terminalul lung (prin el intră curentul); catodul este cel scurt (prin el curentul iese, spre GND).
3. 3. GND este pinul de întoarcere (masa), borna − a plăcii. micro:bit furnizează aproximativ 3,3 V (pinul 3V), iar Arduino 5 V (pinul 5V).

MODULUL 1 · ANEXA ELEVULUI

Răspunsuri la „3 întrebări rapide”

Lecția 1.4

1. 1. R = (Uplacă − ULED) ÷ I.
2. 2. Pentru siguranță: o rezistență puțin mai mare protejează LED-ul; costul este doar o lumină puțin mai slabă.
3. 3. micro:bit → 220 Ω; Arduino (la ținta de 6 mA) → 560 Ω.

Lecția 1.5

1. 1. Alimentare (3V / 5V) → rezistor de protecție → anod → catod → GND.
2. 2. Nu se aprinde deloc, dar nu se deteriorează: LED-ul permite trecerea curentului într-un singur sens. Semnul distinctiv: terminalul lung nu este orientat spre rezistor.
3. 3. Unitatea de măsură a rezistorului: kΩ în loc de Ω; se verifică în caseta de proprietăți.

Lecția 1.6

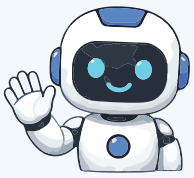
1. 1. Pinul 3V / 5V furnizează curent permanent și nu poate fi comandat; doar pinii comandați (0, respectiv 13) execută instrucțiunile programului.
2. 2. Repetă la nesfârșit instrucțiunile din interiorul ei; în blocuri se numește forever, în Python se scrie while True: (în varianta Tinkercad, basic.forever), iar în C++ este funcția void loop().
3. 3. LED-ul clipește de 5 ori mai repede: fiecare stare durează 100 ms în loc de 500 ms.

MOTOARE
și actuatori

LECȚIA 2.1

Motorul de curent continuu: prima mișcare

Robotul tău știe deja să lumineze. Din această lecție, învață să miște. Vei face cunoștință cu familia actuatorilor și vei porni primul motor, fără niciun rând de cod.



CE VEI ȘTI LA FINALUL LECȚIEI?

- Să definești actuatorul și să oferi exemple din viața de zi cu zi
- Să descrii principiul prin care motorul transformă energia electrică în mișcare de rotație
- Să inversezi sensul de rotație al unui motor și să explici de ce metoda funcționează

SECȚIUNEA 01 · TEORIE

Actuatorii: partea care acționează

DEJA CUNOȘTI (DIN MODULUL 1)

Robotul simte → decide → acționează. Curentul circulă doar în circuit închis, pus în mișcare de tensiune, iar LED-ul transformă energia electrică în lumină. Astăzi ne ocupăm de a treia funcție: acțiunea.

Orice componentă care transformă energia electrică într-un efect fizic (mișcare, lumină, sunet, căldură) se numește **element de execuție** sau **actuator** (termen întâlnit încă din Lecția 1.1). LED-ul din Modulul 1 a fost, fără să știi, primul tău actuator: transformă energia electrică în lumină. Difuzorul telefonului transformă energia electrică în sunet. Iar componenta centrală de astăzi, **motorul de curent continuu**, transformă energia electrică în mișcare de rotație: el rotește roțile aspiratorului robot, ventilatorul calculatorului și periuța de dinți electrică.

Cum apare rotația?

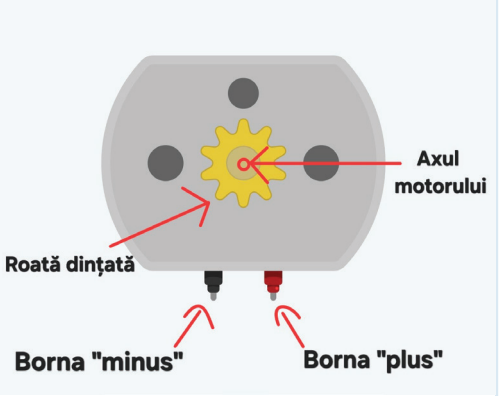
Cunoști de la fizică efectele curentului electric; printre ele, efectul magnetic: un conductor parcurs de curent electric generează în jurul său un câmp magnetic. În motor, curentul trece printr-o bobină (un conductor înfășurat în mai multe spire), care devine astfel un electromagnet. Bobina este montată pe partea mobilă a motorului (rotorul) și se află între doi magneți permanenți ficși (statorul). Magneții atrag și resping electromagnetul, iar axul se rotește. O piesă din interior, numită colector (comutator), schimbă sensul curentului prin bobină la fiecare jumătate de rotație, exact la momentul potrivit pentru ca împingerea să continue, rotație după rotație.

↔ **De aici vine și regula sensului: dacă inversezi polaritatea (schimbi firele între + și −), motorul se rotește invers. O vei verifica practic peste câteva minute.**

SECȚIUNEA 02 · PIESELE DE AZI

Piesă nouă: Motorul de curent continuu (DC Motor)

Piesă nouă: Motorul de curent continuu (DC Motor) PRIMA APARIȚIE



CE ESTE?

Un cilindru metalic cu un ax care iese printr-un capăt și două borne de contact. În interior se află bobina (rotorul), doi magneți permanenți (statorul) și colectorul. „Curent continuu” (prescurtat DC, din limba engleză: direct current) înseamnă curent care circulă mereu în același sens, exact cel furnizat de baterii.

CE FACE?

Transformă energia electrică în energie mecanică de rotație. Viteza de rotație depinde de tensiunea aplicată; sensul de rotație depinde de polaritate.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

Jucării cu baterii, ventilatoare, periuțe de dinți electrice, geamurile electrice ale mașinii, aspiratorul robot.

DE CE NE TREBUIE AZI?

Este cel mai simplu mod de a observa energia electrică producând mișcare; în Lecția 2.3 îl vom comanda din program.

Ruda lui din telefon: motorul cu vibrații

Vibrația telefonului este determinată tot de un motor de curent continuu, de dimensiuni foarte mici, cu o mică masă montată excentric (în afara axei de rotație) pe ax. Când axul se rotește rapid, masa dezechilibrată scutură întregul telefon. Componenta există și în Tinkercad, sub denumirea „Vibration Motor”; o poți încerca la finalul lecției.

SECȚIUNEA 03 · PRACTICĂ

Primul motor care se rotește

CIRCUITUL DEMO AL LECȚIEI

▶ Deschide circuitul „Primul motor”

Baterie → întrerupător → motor, construit în prealabil. Apasă „Copy and Tinker” și lucrează în propria copie.

SECȚIUNEA 03 · PRACTICĂ

Primul motor care se rotește

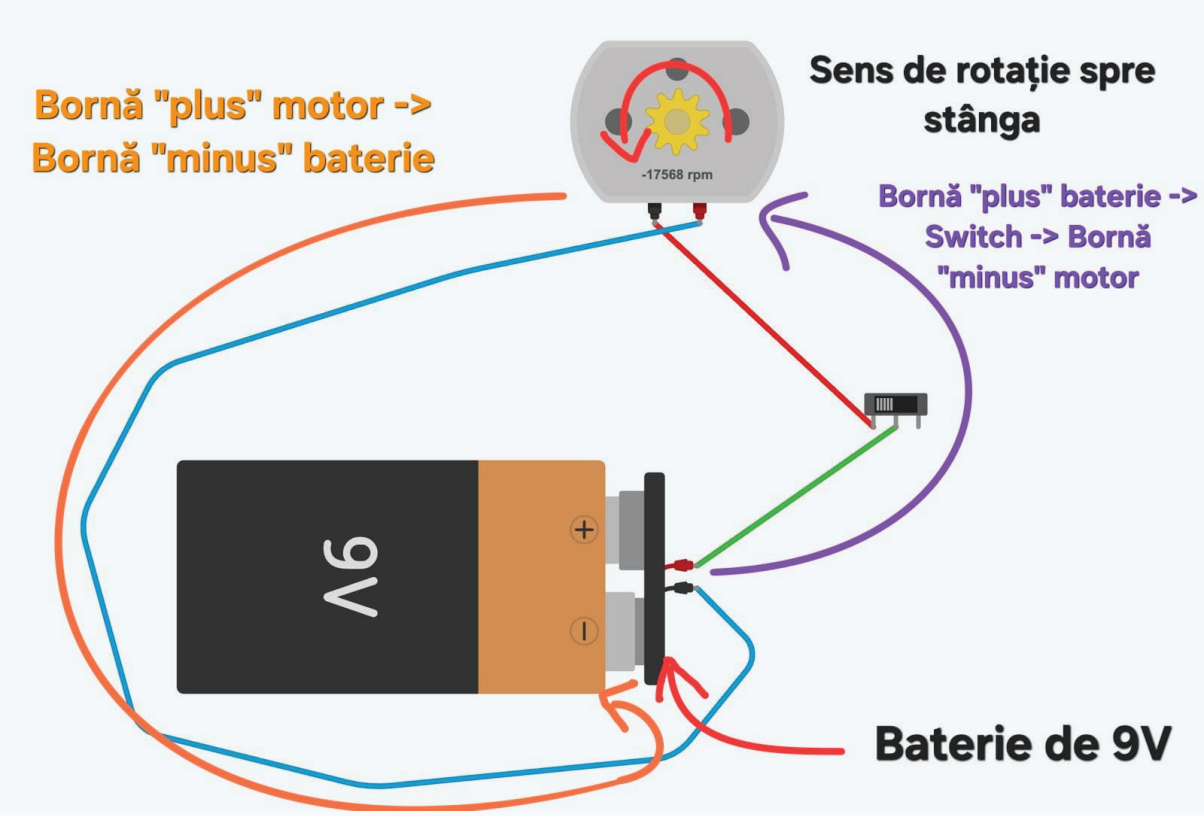
1 Pornește simularea și citește turația

Apasă „Start Simulation”, apoi închide întrerupătorul. Motorul pornește, iar simulatorul afișează lângă el turația: numărul de rotații pe minut, prescurtat rot/min (în simulator apare ca RPM, din limba engleză: revolutions per minute). Notează valoarea în caiet; o vei compara cu o altă măsurătoare în lecția viitoare.

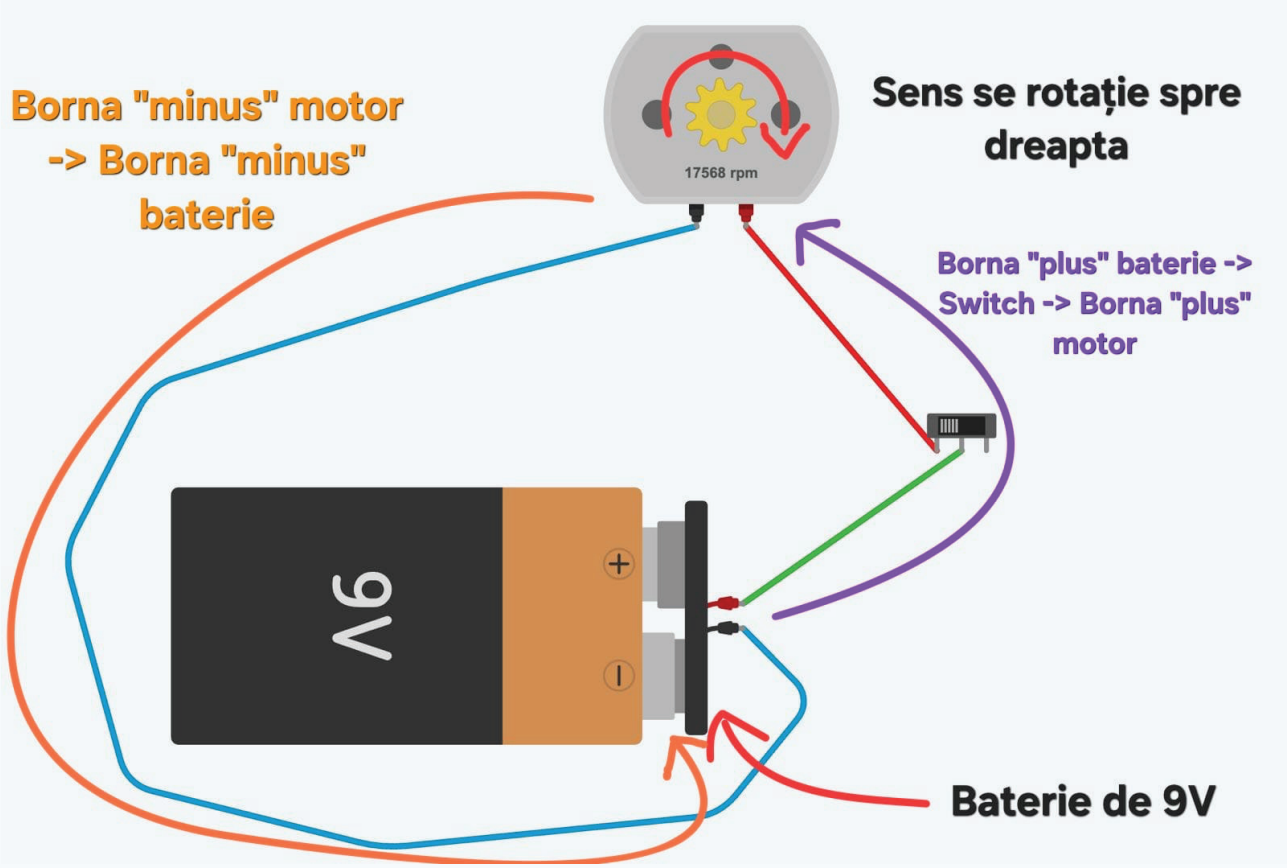
☐ **Motorul se rotește și am notat turația**

2 Inversează sensul de rotație

Oprește simularea. Scoate cele două fire din bornele motorului și conectează-le invers (firul care era la borna de sus intră acum la cea de jos, și invers). Pornește din nou: săgeata de rotație s-a inversat, aceeași viteză, sens opus. Ai schimbat polaritatea, exact ca în teorie.



SECȚIUNEA 03 · PRACTICĂ



☐ **Am văzut motorul rotindu-se în ambele sensuri**

⚠ Un detaliu cinstit despre motoare și baterii

Motorul absoarbe un curent mult mai mare decât un LED (sute de miliamperi față de câțiva miliamperi); din acest motiv, jucăriile cu motoare descarcă bateriile rapid. Reține observația: devine importantă în Lecția 2.3, când vom conecta motorul la placă.

↔ **Polaritate inversată = sens de rotație inversat.**

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

„Actuator sau senzor?”

Cum se joacă? (8 minute, toată clasa)

Pe ecran (sau numite de profesor) apar, pe rând, 10 componente și aparate: motorul ștergătoarelor de parbriz, camera telefonului, becul din tavan, difuzorul, termometrul de cameră, motorul cu vibrații al telefonului, microfonul, încuietoarea electrică a ușii, senzorul de parcare, ecranul tactil. Clasa votează: actuator sau senzor? Regula deciziei: senzorul culege informații din mediu, actuatorul modifică ceva în mediu. Capcana finală: ecranul tactil este și una, și alta: afișează (actuator) și detectează atingerea (senzor).

Varianta Kahoot: cele 10 componente, sub formă de întrebări cu imagini; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 2.1-J: un tabel cu cele 10 componente și două coloane de bifat: simte / acționează

Varianta fără ecrane: profesorul numește componenta; mâna stângă sus = senzor, mâna dreaptă = actuator, amândouă = ambele; cine greșește explică de ce

Provocare: pentru cei rapizi

Așază pe planșa de lucru două motoare. Conectează-le întâi în paralel la aceeași baterie (fiecare motor direct la borne), apoi în serie (unul după altul, în același circuit). Compară turațiile afișate în cele două cazuri și explică diferența folosind tensiunea. Indiciu: în serie, cei doi consumatori își împart tensiunea bateriei.

3 întrebări rapide

1. Ce este un actuator? Dă două exemple din casa ta.
2. Ce se întâmplă cu motorul dacă inversezi firele la borne și de ce?
3. Ce mărime afișează simulatorul lângă motor și în ce se măsoară?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

actuator: element de execuție; componentă care transformă energia electrică într-un efect fizic: mișcare, lumină, sunet

motor de curent continuu (DC): motor care transformă energia electrică în mișcare de rotație; sensul depinde de polaritate

bobină: conductor înfășurat în mai multe spire; parcurs de curent, devine electromagnet

colector: piesa (comutatorul) care schimbă sensul curentului prin bobină, pentru ca rotația să continue

polaritate: modul de conectare a firelor la bornele + și − ale sursei

turație (rot/min): numărul de rotații pe minut; în simulator apare ca RPM

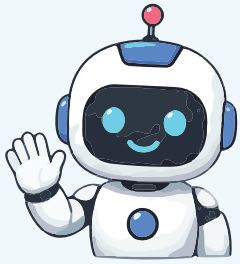
Ce trebuie să știi după această lecție?

- ✓ Actuatorul este componenta care „acționează” din structura robotului; LED-ul a fost primul tău actuator
- ✓ Polaritate inversată = sens de rotație inversat
- ✓ Motorul DC se rotește deoarece bobina parcursă de curent devine electromagnet între magneții permanenți ficși
- ✓ Turația se măsoară în rotații pe minut și se citește direct în simulator

LECȚIA 2.2

Roți dințate și reductoare: viteză sau forță

Motorul din lecția trecută se rotește amețitor de repede, dar are o forță foarte mică. În această lecție vei afla cum schimbă roțile dințate viteza pe forță și vei măsura totul în simulator.



CE VEI ȘTI LA FINALUL LECȚIEI?

- Să definești turația și cuplul și să explici diferența dintre ele
- Să calculezi raportul de transmisie al unui angrenaj din numărul de dinți
- Să compari în simulator motorul simplu cu motoreductorul și să explici diferența

SECȚIUNEA 01 · TEORIE

Turație și cuplu: repede sau puternic

DEJA CUNOȘTI (DIN LECȚIA 2.1)

Motorul DC transformă energia electrică în rotație, iar turația (rot/min) se citește în simulator. Ai notat o valoare; o folosim astăzi.

Un motor mic de jucărie se rotește cu mii de rotații pe minut, dar, dacă ai pune (imaginar) degetul pe ax, s-ar opri imediat. Viteza sa de rotație se numește turație (notată “n”); capacitatea sa de a răsuci axul, adică momentul forței pe care îl dezvoltă, se numește cuplu (notat M și măsurat în Newton·metri, N·m). Cele două mărimi sunt în echilibru: la aceeași putere a motorului, nu le poți avea pe amândouă la maximum. Ai întâlnit exact acest principiu la pârghiile studiate la fizică: cu un braț mai lung apeși cu o forță mai mică, dar mâna parcurge un drum mai lung.

Schimbul „viteză contra forță” este realizat de roțile dințate: roți cu dinți care se îmbucă exact, astfel încât una o rotește pe cealaltă fără alunecare. Când o roată mică (numită roată conducătoare, fixată pe axul motorului) rotește o roată mare (roata condusă), roata mare se rotește mai încet, dar cu un cuplu mai mare. Un ansamblu de roți dințate care lucrează împreună se numește angrenaj; un ansamblu de angrenaje închis într-o carcasă, care reduce turația, se numește reductor.

Formula lecției: raportul de transmisie

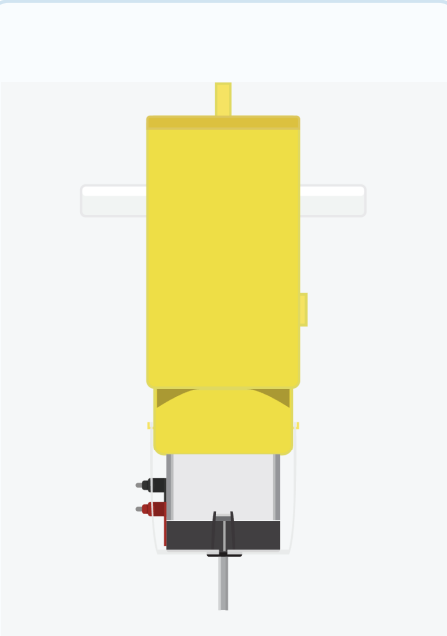
i = numărul de dinți al roții conduse ÷ numărul de dinți al roții conducătoare

Turația scade de “i” ori, iar cuplul crește de aproximativ “i” ori (spunem „aproximativ” deoarece o mică parte din energie se pierde prin frecare). Exemplu: roata conducătoare a motorului are 12 dinți, iar roata condusă are 60 → i = 60 ÷ 12 = 5. Un motor cu 3 000 rot/min iese din angrenaj cu 3 000 ÷ 5 = 600 rot/min, dar cu un cuplu de circa 5 ori mai mare. Este o proporție, exact matematica pe care o cunoști din gimnaziu.

Piesă nouă: Motoreductorul

Piesă nouă: Motoreductorul

PRIMA APARIȚIE



CE ESTE?

Un motor DC și un reductor cu roți dințate, montate în aceeași carcasă, de obicei galbenă, cu axul de ieșire în lateral. În Tinkercad îl găsești sub denumirea englezească „Hobby Gearmotor”.

CE FACE?

Reduce turația motorului și îi crește cuplul, exact în raportul angrenajelor din interior. La ieșire: rotație mai lentă, dar cu un cuplu suficient pentru a urni roțile unui robot.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

Aproape orice robot pe roți de la concursurile școlare, jucării motorizate, mecanismele imprimantelor obișnuite.

DE CE NE TREBUIE AZI?

Pentru a observa, cu valori concrete, efectul unui reductor: aceeași baterie, aceeași tensiune, turație complet diferită.

SECȚIUNEA 02 · PRACTICĂ

Măsurăm: motor simplu contra motoreductor

CIRCUITUL DEMO AL LECȚIEI

▶ **Deschide circuitul „Motor și motoreductor”**

Aceeași baterie și același întrerupător alimentează, în paralel, un motor simplu și un motoreductor.

SECȚIUNEA 02 · PRACTICĂ

Măsurăm: motor simplu contra motoreductor

1 Pornește și citește ambele turații

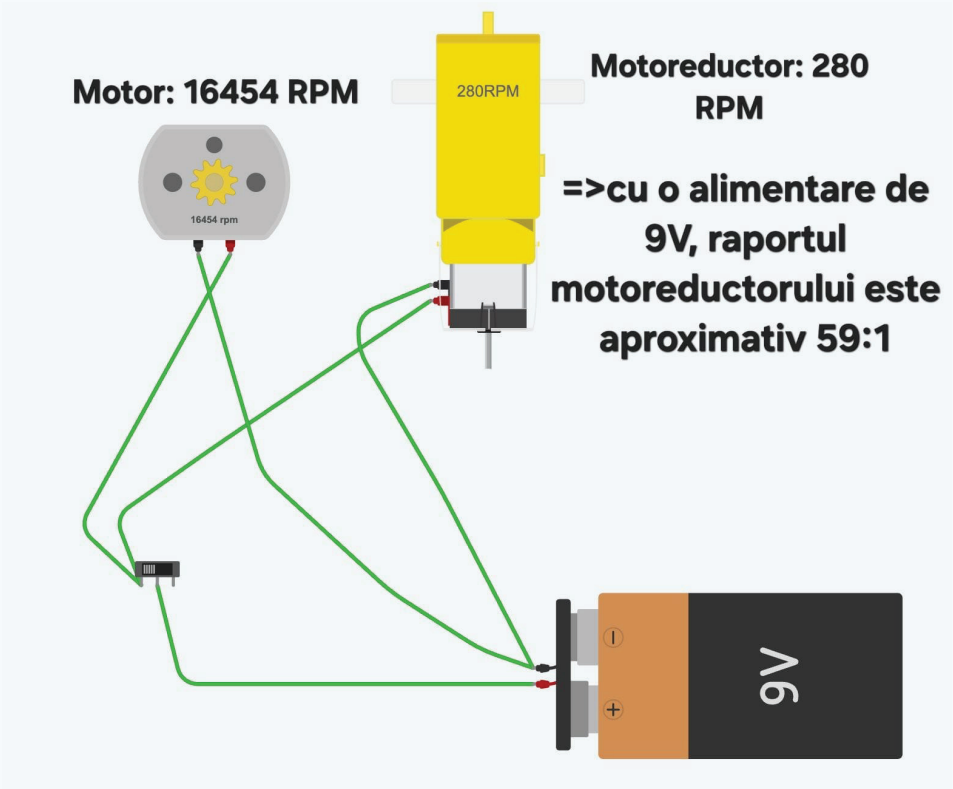
Apasă „Start Simulation” și închide întrerupătorul. Notează în caiet turația afișată de fiecare: motorul simplu indică mii de rot/min, iar motoreductorul sute sau chiar zeci. Aceeași sursă, aceeași tensiune. Motorul simplu ar trebui să indice aceeași turație ca în lecția trecută; compar-o cu valoarea notată atunci. Notează acum și turația motoreductorului.

☐ **Am notat cele două turații**

2 Calculează raportul reductorului

Împarte turația motorului simplu la turația motoreductorului: rezultatul este chiar raportul de transmisie al reductorului din carcasă. Scrie calculul complet, nu doar rezultatul.

☐ **Am calculat raportul**



SECȚIUNEA 03 · EXERSĂM

Rezolvă pe caiet, cu formula în față. Scrie calculul complet.

Exercițiul 1 · Roata conducătoare are 10 dinți, roata condusă are 40. Motorul intră cu 2 400 rot/min. Ce turație iese?

Exercițiul 2 · Vrei să cobori 6 000 rot/min la 1 500 rot/min. Ce raport de transmisie este necesar? Propune o pereche de roți dințate care îl obține.

Exercițiul 3 (invers) · Bicicleta ta: pedalezi cu 60 de rotații pe minut, iar roata din spate se rotește cu 180. Aici turația a crescut: ce s-a întâmplat, în schimb, cu forța? Unde simți acest lucru când urci o pantă?

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă ? (8 minute, în perechi)

Doi elevi devin roți dințate angrenate. Profesorul anunță raportul, de exemplu „3 la 1”: elevul-roată-mică bate din palme de 3 ori în timp ce elevul-roată-mare bate o singură dată, în ritm constant. Clasa verifică sincronizarea și strigă raportul. Se schimbă perechile și rapoartele (2:1, 4:1, 5:2 pentru curajoși). Întrebarea de final, discutată împreună: care dintre cei doi „obosește” mai repede și ce mărime din lecție reprezintă efortul lui?

Varianta Kahoot: perechi de roți dințate desenate, cu numărul de dinți vizibil; elevii calculează raportul contra cronometru; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 2.2-J: angrenaje desenate; completează turația lipsă și sensul de rotație al fiecărei roți

Varianta fără ecrane: chiar jocul principal; necesită doar palme și ritm

Provocare: întrebarea de detectiv

Două roți dințate îmbucate se rotesc mereu în sensuri opuse. Dacă vrei ca roata de ieșire să se rotească în același sens cu motorul, câte roți trebuie să pui în lanț? Desenează răspunsul în caiet.

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

3 întrebări rapide

1. Definește, în cuvinte proprii, turația și cuplul.
2. Roata conducătoare are 15 dinți, iar cea condusă 45. Care este raportul de transmisie și ce se întâmplă cu turația?
3. De ce robotul pe roți folosește motoreductor, nu motor simplu?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

cuplu: momentul forței cu care motorul rotește axul (M); se măsoară în Newton·metri (N·m)

roată dințată: roată cu dinți care se îmbucă exact cu altă roată, transmițând rotația fără alunecare

angrenaj: două sau mai multe roți dințate care lucrează împreună

reductor: ansamblu de angrenaje care reduce turația și crește cuplul

raport de transmisie (i): numărul de dinți ai roții conduse împărțit la numărul de dinți ai roții conducătoare

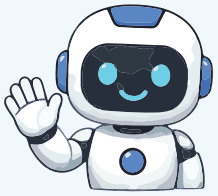
motoreductor: motor DC și reductor în aceeași carcasă

Ce trebuie să știi după această lecție?

- ✓ Turația (n) arată cât de repede; cuplul (M) arată cât de puternic; la aceeași putere, unul crește doar dacă scade celălalt
- ✓ i = dinți conduși ÷ dinți conducători; turația scade de “i” ori, cuplul crește de circa “i” ori
- ✓ Motoreductorul este motorul + reductorul în aceeași carcasă; alegerea firească pentru roboții pe roți
- ✓ Raportul unui reductor necunoscut se află împărțind turația de intrare la cea de ieșire

LECȚIA 2.3 · 50 DE MINUTE

Tranzistorul: motorul comandat din program



În această lecție se întâlnesc Modulul 1 și Modulul 2: programul care făcea LED-ul să clipească va porni și va opri un motor. Între ele este nevoie de o singură componentă nouă: tranzistorul.

CE VEI ȘTI LA FINALUL LECȚIEI?

- Să explici de ce motorul nu se conectează niciodată direct la un pin al plăcii
- Să descrii cum comandă un curent mic (de la pin) un curent mare (prin motor), prin intermediul tranzistorului
- Să construiești și să programezi, pe traseul tău, un motor care pornește și se oprește singur

SECȚIUNEA 01 · TEORIE

De ce motorul nu intră direct în pin?

DEJA CUNOȘTI (DIN LECȚIILE 1.6 ȘI 2.1)

Pinii 0, 1, 2 ai plăcii se deschid și se închid din program cu pins.digital_write_pin(DigitalPin.P0, 1). Iar motorul, după cum ai văzut, absoarbe un curent mare, mult mai mare decât un LED.

Pinii plăcii sunt porți de comandă, nu de forță: printr-un pin de micro:bit pot trece în siguranță doar câțiva miliamperi, iar printr-un pin de Arduino, aproximativ 20 mA (cu limita absolută de 40 mA). Un motor de jucărie cere o sută sau câteva sute de miliamperi, iar la pornire chiar mai mult. Conectat direct la pin, motorul ar cere un curent de zeci de ori peste limită și ar putea distruge pinul definitiv. De aici, regula de aur a modulului: motoarele nu se conectează niciodată direct la un pin.

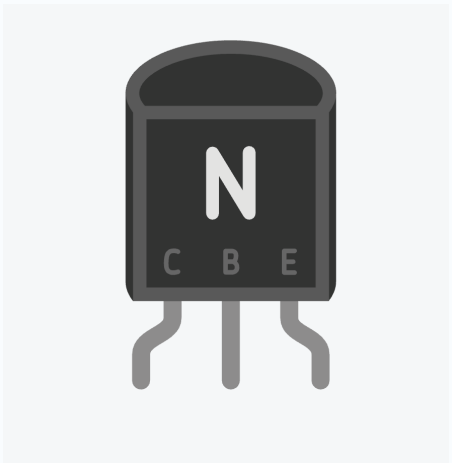
Soluția este veche și elegantă: pinul nu alimentează motorul, ci doar transmite comanda unei componente care poate conduce un curent mare: tranzistorul. Gândește-te la robinetul de apă din baie: cu două degete (efort mic) deschizi un drum prin care trece multă apă (efect mare). Tranzistorul este exact acest lucru, în variantă electronică: un comutator fără piese mobile, deschis de un curent foarte mic.

SECȚIUNEA 02 · PIESELE DE AZI

Piesă nouă: Tranzistorul (tip NPN)

Piesă nouă: Tranzistorul (tip NPN)

PRIMA APARIȚIE



CE ESTE?

O componentă semiconductoare mică, neagră, cu trei terminale: baza (B), intrarea de comandă, respectiv colectorul (C) și emitorul (E), capetele traseului de forță. (Numele „colector” coincide întâmplător cu cel al piesei din motorul de la Lecția 2.1; sunt componente diferite.) Este cea mai importantă invenție a electronicii: procesorul telefonului tău conține miliarde de tranzistori microscopici.

CE FACE?

Un curent foarte mic, care circulă de la bază spre emitor, deschide traseul principal de la colector spre emitor. Curent mic la bază = poartă deschisă; curent zero la bază = poartă închisă. Comanda este dată de pin, printr-un rezistor de protecție de 1 kΩ.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În orice aparat electronic, de la telecomandă la laptop. Este „cărămida” din care sunt construite toate „creierele” electronice.

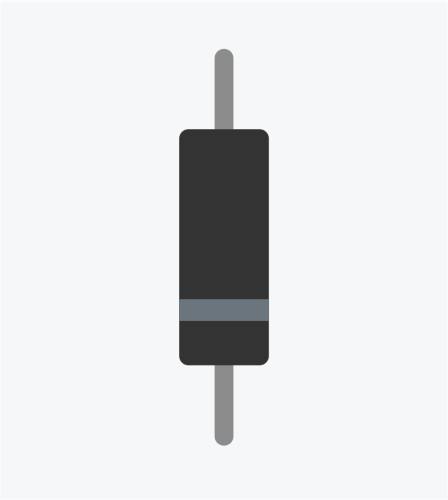
DE CE NE TREBUIE AZI?

Este veriga lipsă dintre pinul cu putere limitată și motorul cu consum ridicat: pinul comandă tranzistorul, iar tranzistorul conduce curentul motorului.

SECȚIUNEA 02 · PIESELE DE AZI

Piesă nouă: Dioda

Piesă nouă: Dioda — PRIMA APARIȚIE



CE ESTE?

O componentă semiconductoare cu două terminale, care permite trecerea curentului într-un singur sens; cunoști această regulă de la LED, care este, de fapt, o diodă luminiscentă. Sensul este indicat de inelul (dunga) de pe capsulă: dunga marchează catodul, capătul de ieșire.

CE FACE AICI: DIODĂ DE PROTECȚIE?

Bobina motorului se opune opririi bruște a curentului: în momentul întreruperii, ea generează un scurt impuls de tensiune (fenomen numit autoinducție), asemănător „loviturii de berbec” din instalațiile de apă, bufnitura din perete când închizi robinetul prea repede. Dioda, montată invers în paralel cu motorul, oferă impulsului un traseu scurt și sigur de descărcare, protejând astfel tranzistorul.

UNDE O ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În orice încărcător și orice sursă de alimentare; câte una de protecție stă lângă aproape fiecare motor și releu din electronica modernă.

DE CE NE TREBUIE AZI?

Fără ea, fiecare oprire a motorului ar reprezenta un mic șoc de tensiune pentru tranzistor. Cu ea, circuitul funcționează în siguranță.

SECȚIUNEA 03 · PRACTICĂ

Schema comună: pin → tranzistor → motor

Cele două drumuri

Tranzistorul este un robinet. Prin el trec două trasee complet diferite:

Drumul de forță : curentul mare care rotește motorul. Pleacă din alimentare, trece prin motor și prin tranzistor spre GND. Nu intră niciodată în placă.

Drumul de comandă : curentul mic care vine din pin și îi transmite tranzistorului doar „deschide-te” sau „închide-te”. Este mâna care răsucește robinetul, nu apa.

SECȚIUNEA 03 · PRACTICĂ

Schema comună: pin → tranzistor → motor

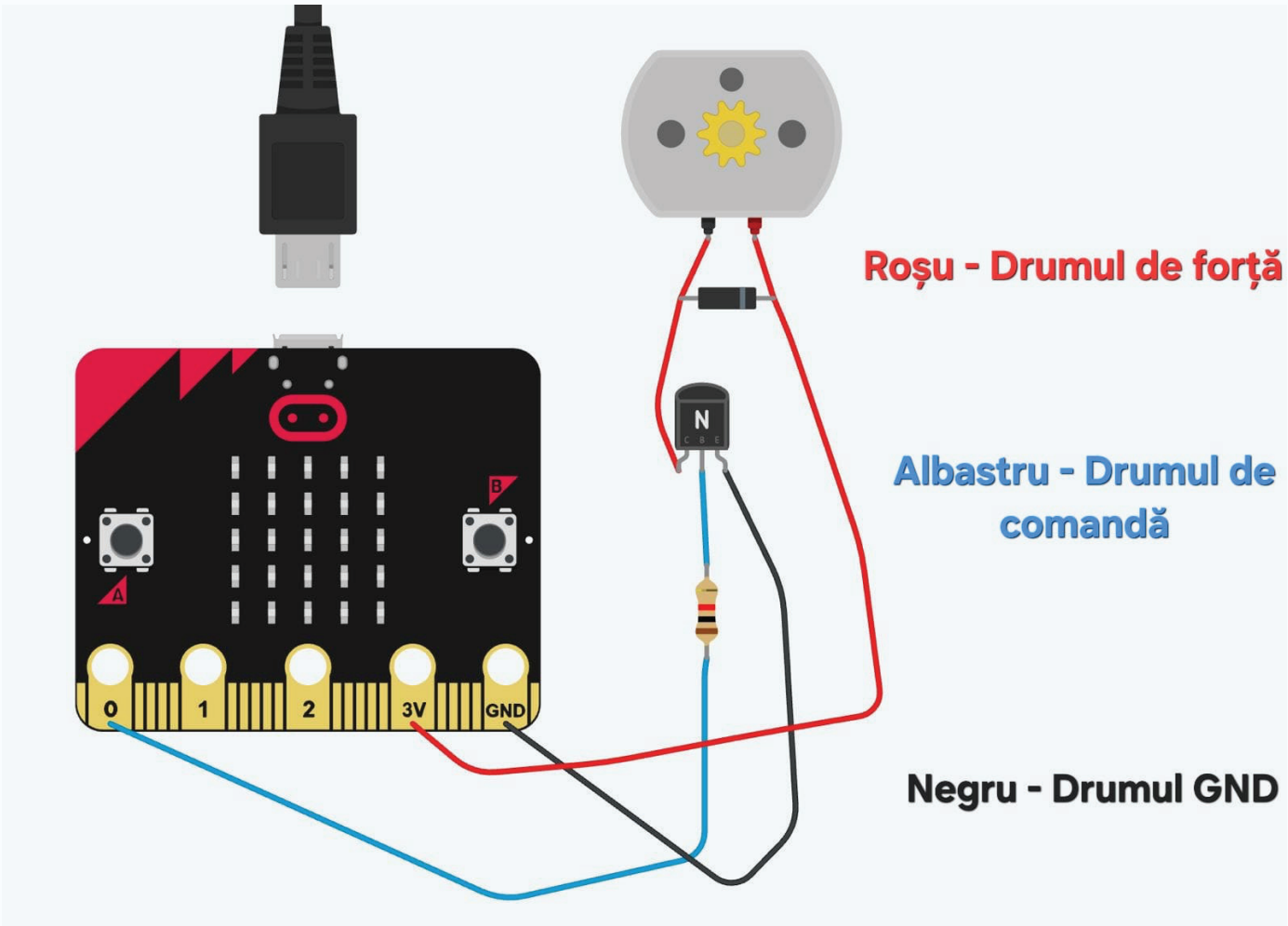


Fig. 1. O singură schemă, două plăci. Se schimbă doar numele pinului și tensiunea de alimentare.

TRASEUL A · MICRO:BIT

Pinul de comandă: P0 · Alimentarea motorului: pinul 3V

CIRCUIT DE START

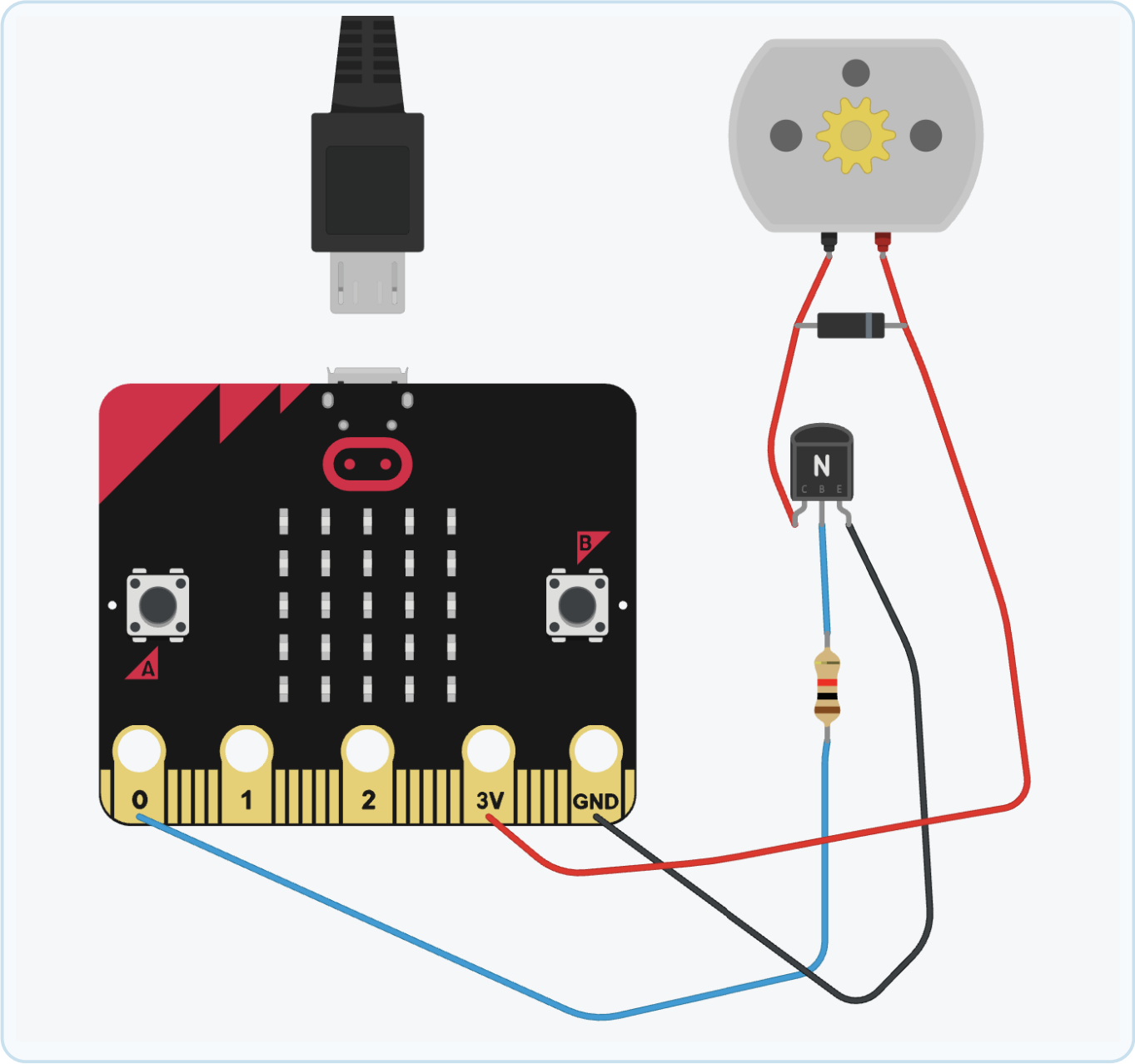
▶ Deschide circuitul de start 2.3 · Micro:bit

Placa, motorul, tranzistorul, rezistorul de 1 kΩ și dioda se află pe planșa de lucru, neconectate. Apasă „Copy and Tinker”.

TRASEUL A · MICRO:BIT
Pinul de comandă: P0 · Alimentarea motorului: pinul 3V

1 Cablează după schemă

Trasează firele exact ca în Fig. 1: P0 → rezistor 1 kΩ → bază; emitor → GND; 3V → motor → colector; dioda în paralel cu motorul, cu dunga spre 3V. Compară planșa ta cu captura înainte de a merge mai departe.

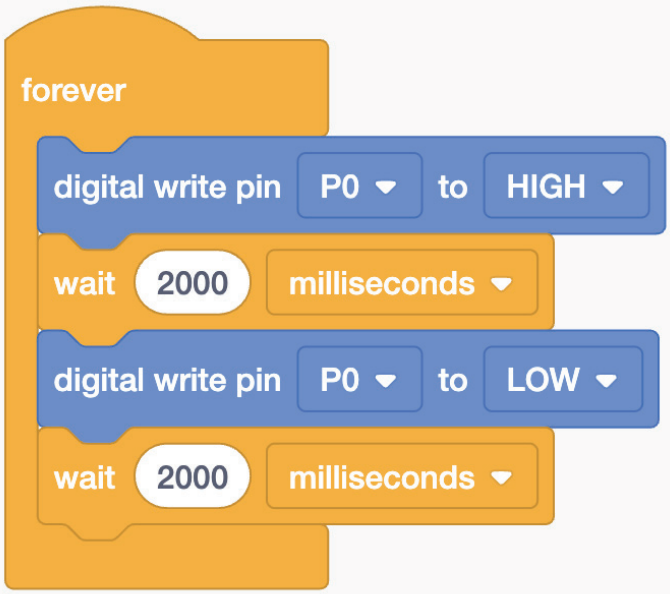


☐ **Circuitul arată ca în captură**

TRASEUL A · MICRO:BIT
Pinul de comandă: P0 · Alimentarea motorului: pinul 3V

2 Programează: întâi blocuri, apoi Python

Apasă „Code”. În Blocuri, construiește bucla: forever → digital write pin P0 to HIGH → wait 2000 ms → digital write pin P0 to LOW → wait 2000 ms. Rulează: motorul funcționează 2 secunde, apoi stă 2 secunde. Comută apoi pe Python și scrie același program ca text; îl găsești mai jos, rând cu rând.



☐ **Motorul pornește și se oprește singur**

VARIANTA ÎN TEXT · ACELEAȘI INSTRUCȚIUNI — **PYTHON · MICRO:BIT**

```
# Lecția 2.3: motor pornit și oprit din program
# (varianta Tinkercad / MakeCode)

def on_forever():   # definim un „bloc de instrucțiuni” cu numele on_forever
    pins.digital_write_pin(DigitalPin.P0, 1)   # 1 pe pinul P0 → tranzistorul se deschide,
motorul pornește
    basic.pause(2000)   # merge 2 secunde
    pins.digital_write_pin(DigitalPin.P0, 0)   # 0 pe pinul P0 → tranzistorul se închide,
motorul se oprește
    basic.pause(2000)   # pauză 2 secunde

basic.forever(on_forever)   # spune plăcii: repetă la nesfârșit blocul de mai sus
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Pinul de comandă: 9 · Alimentarea motorului: pinul 5V

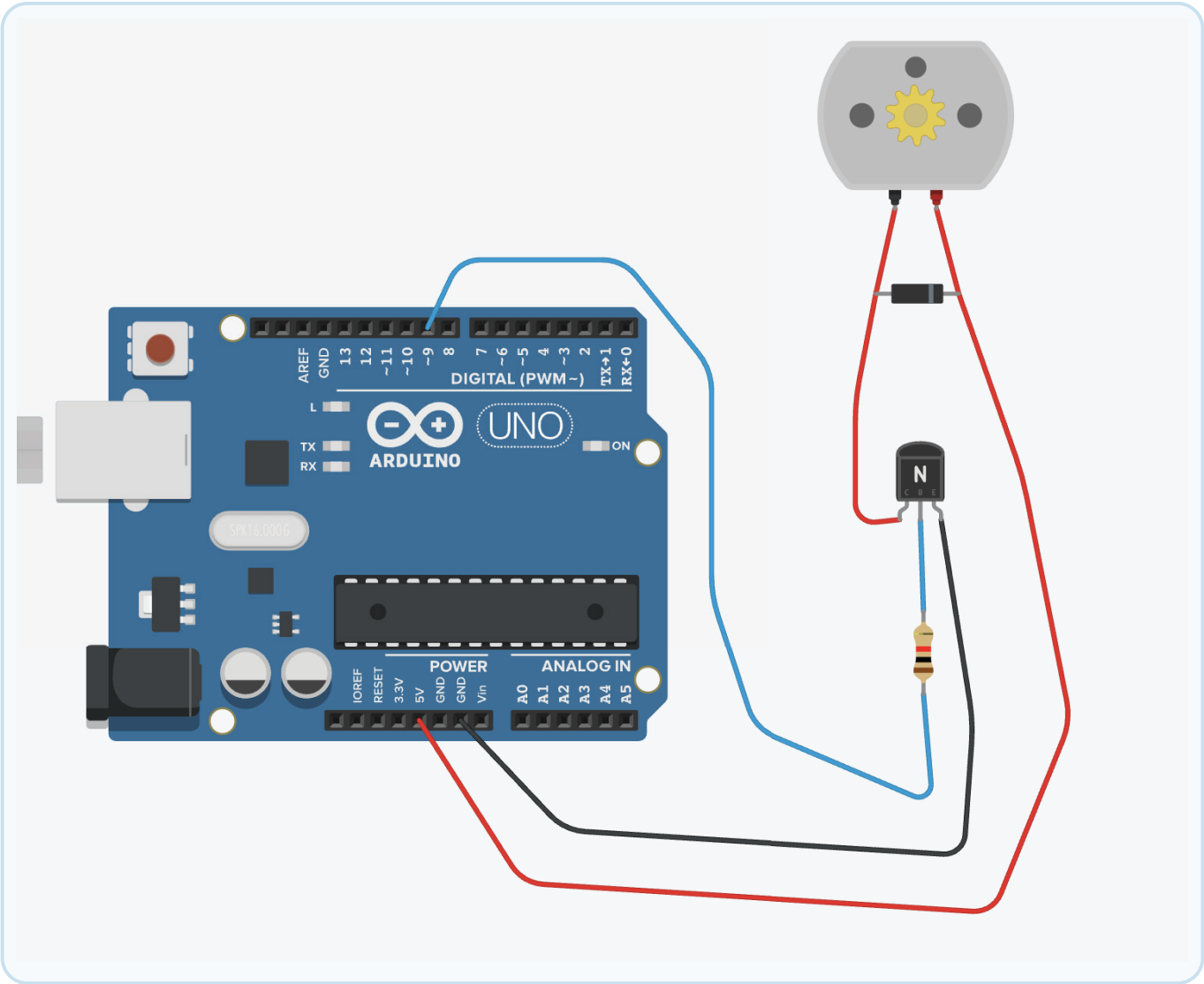
CIRCUIT DE START

▶ Deschide circuitul de start 2.3 · Arduino

Aceleași componente, placa Arduino în locul plăcii Micro:bit. Apasă „Copy and Tinker”.

1 Cablează după aceeași schemă

Pinul 9 → rezistor 1 kΩ → bază; emitor → GND; 5V → motor → colector; dioda în paralel cu motorul, cu dunga spre 5V. Observă: schema este identică, doar numele pinului și tensiunea diferă.



☐ **Circuitul arată ca în captură**

TRASEUL B · ARDUINO UNO

Pinul de comandă: 9 · Alimentarea motorului: pinul 5V

2 Programează în blocuri și privește vitrina C++

Apasă „Code” și alege modul „Blocks + Text”. Construiește: forever → set pin 9 to HIGH → wait 2000 ms → set pin 9 to LOW → wait 2000 ms. În dreapta, Tinkercad scrie automat codul C++ echivalent: aceleași cinci instrucțiuni, în limbajul în care se programează Arduino în fabrici și ateliere. Doar privește-l; nu trebuie învățat.

```
// C++ code
//
void setup()
{
  pinMode(9, OUTPUT);
}

void loop()
{
  digitalWrite(9, HIGH);
  delay(2000); // Wait for 2000 millisecond(s)
  digitalWrite(9, LOW);
  delay(2000); // Wait for 2000 millisecond(s)
}
```

☐ **Motorul pornește și se oprește singur**

C++ generat, cu comentarii în limba română — **C++ · ARDUINO**

```
void setup() // se execută o singură dată, la pornire
{
  pinMode(9, OUTPUT);   // pinul 9 devine pin de ieșire
}

void loop() {   // se repetă la nesfârșit
  digitalWrite(9, HIGH);   // HIGH = 1 → motorul pornește
  delay(2000);   // delay = sleep: pauză în milisecunde
  digitalWrite(9, LOW);   // LOW = 0 → motorul se oprește
  delay(2000); // pauză două secunde
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT	ARDUINO UNO
Pinul suportă doar câțiva miliamperi: cel mai strict dintre cei doi. Python + blocuri, ca de obicei.	Pinul suportă aproximativ 20 mA: mai tolerant, dar tot de zeci de ori sub necesarul motorului. Blocuri + vitrina C++.

Concluzia: niciuna dintre plăci nu poate alimenta un motor din pin. Tranzistorul este obligatoriu pe ambele trasee; diferă doar cât de repede ai distruge pinul fără el.

SECȚIUNEA 04 · DEPANARE

⚠ Motorul nu pornește deloc?

Verifică în ordine: 1) Rezistorul de 1 kΩ se află pe traseul spre bază (nu spre colector)? 2) Emitorul ajunge exact la GND? 3) Traseul de forță este complet: alimentare → motor → colector → emitor → GND? 4) Codul scrie pe pinul corect (P0, respectiv 9)? 5) Simularea este pornită?

⚠ Greșeala periculoasă a lecției: dioda montată invers

Cu dunga spre GND în loc de alimentare, dioda devine un traseu direct: scurtcircuit în clipa în care tranzistorul se deschide. În simulator primești doar o avertizare; în realitate, ar mirosi a plastic ars. Regula de reținut: dunga diodei privește spre alimentare.

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă? (6 minute, toată clasa)

Patru roluri, jucate în fața clasei: Pinul (un elev cu o lanternă de telefon), Tranzistorul (un elev cu brațele întinse, poarta), Curentul de forță (un șir de 4–5 elevi) și Motorul (un elev care se rotește pe loc). Regula: șirul poate trece doar când poarta este deschisă, iar Motorul se rotește doar cât timp trec elevi prin poartă. Pinul aprinde lanterna → Tranzistorul deschide brațele → șirul curge → Motorul se rotește. Lanterna se stinge → poarta se închide → totul se oprește. Clasa observă în direct: semnalul este minuscul, efectul este mare, iar niciun elev din șir nu trece vreodată prin Pin.

Varianta Kahoot: întrebări „ce se întâmplă dacă...?” pe schema lecției; linkul și codul QR se află în folderul lecției

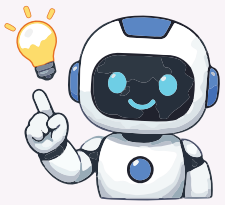
Varianta pe foaie: fișa 2.3-J: schema necompletată; elevii trasează cu două culori traseul de comandă și traseul de forță

Varianta fără ecrane: chiar jocul principal; necesită doar o lanternă de telefon și puțin spațiu

Provocare: pentru cei rapizi

Nivel 1 · Panoul de bord. Adaugă un LED cu rezistorul lui de 1 kΩ, conectat la același pin de comandă. De ce 1 kΩ, și nu 220 Ω ca în Modulul 1? Deoarece pinul alimentează acum două ramuri simultan (baza tranzistorului și LED-ul), iar regula de astăzi rămâne valabilă: pinul este poartă de comandă, nu de forță, deci curentul total trebuie menținut mic. LED-ul luminează mai slab, dar ai un martor luminos: LED aprins = motor pornit.

Nivel 2 · Ritmul tău. Schimbă cele două valori de 2000 astfel încât motorul să funcționeze 3 secunde și să stea doar o jumătate de secundă.



3 întrebări rapide

1. De ce nu conectăm motorul direct la un pin, nici măcar la Arduino?
2. Cum reușește un curent mic de la bază să pornească un curent mare prin motor?
3. Ce rol are dioda montată peste motor și încotro privește dunga ei?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

tranzistor: comutator electronic fără piese mobile, deschis de un curent foarte mic

bază · colector · emitor: cele trei terminale: comanda (B) și capetele traseului de forță (C, E)

curent de comandă: curentul mic, de la pin, care deschide tranzistorul

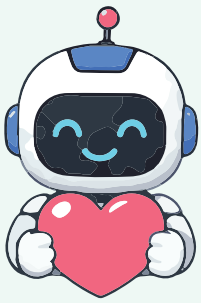
diodă: componentă care permite trecerea curentului într-un singur sens; dunga = catodul

diodă de protecție: diodă montată invers, în paralel cu motorul; absoarbe impulsul de autoinducție de la oprire

HIGH / LOW: denumirile valorilor logice 1 și 0 în mediul Arduino

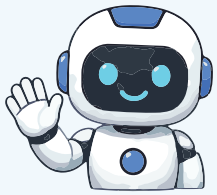
Ce trebuie să știi după această lecție?

- ✓ Pinii sunt porți de comandă, nu de forță: motoarele nu se conectează niciodată direct la pin
- ✓ Tranzistorul: un curent mic la bază deschide traseul principal colector → emitor
- ✓ Dioda de protecție stă în paralel cu motorul, cu dunga spre alimentare, întotdeauna
- ✓ Programul de clipire din Lecția 1.6, neschimbat în logică, comandă acum un motor: aceleași instrucțiuni, alt actuator



LECȚIA 2.4

Schimbarea sensului de rotație: puntea H



Lecția trecută ai pornit motorul din cod. Un robot pe roți trebuie însă să meargă și înainte, și înapoi, fără ca cineva să îi schimbe firele. În această lecție înveți circuitul care face exact acest lucru.

CE VEI ȘTI LA FINALUL LECȚIEI?

- Să explici, pe schema cu patru comutatoare, cum inversează puntea H polaritatea motorului
- Să indici care combinații de comutatoare sunt permise și care produc scurtcircuit
- Să comanzi din program un motor înainte și înapoi, folosind circuitul integrat L293D

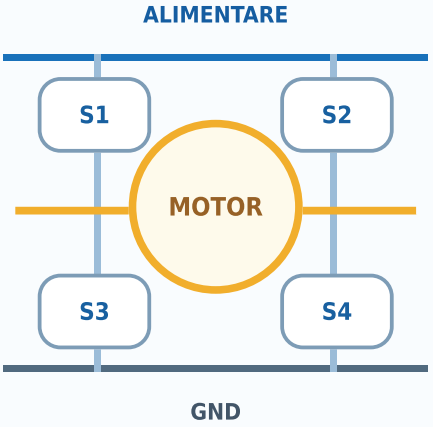
SECȚIUNEA 01 · TEORIE

Patru comutatoare în formă de H

DEJA CUNOȘTI (DIN LECȚIILE 2.1 ȘI 2.3)

Polaritate inversată = sens de rotație inversat. Iar tranzistorul este un comutator comandat din program. Astăzi punem patru asemenea comutatoare să lucreze împreună.

POZA



Desenează motorul în mijloc și câte două comutatoare de fiecare parte: schema seamănă cu litera H, de unde și numele de punte H. Colțurile de sus (S1, S2) vin de la alimentare; colțurile de jos (S3, S4) merg la GND. Închide S1 și S4: curentul intră prin stânga motorului și iese prin dreapta, iar motorul merge înainte. Închide în schimb S2 și S3: curentul traversează motorul exact invers, adică mers înapoi. Aceleași fire, polaritate schimbată doar din comutatoare.

⚠ Combinația interzisă

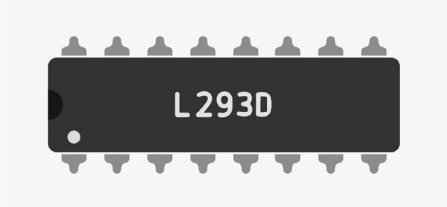
Dacă închizi două comutatoare de pe aceeași parte (S1 cu S3, sau S2 cu S4), curentul găsește drum direct de la alimentare la GND, ocolind motorul: scurtcircuit. Într-o punte construită de tine din tranzistori, acest lucru ar distruge componentele pe loc. Reține regula: se închid doar comutatoare de pe diagonale opuse.

Vestea bună: nu trebuie să construim puntea din patru tranzistori. Electronica modernă o oferă gata construită, sub forma unui circuit integrat , adică un „cip”: mai multe circuite complete, închise într-o capsulă cu terminale (pini). Al nostru se numește L293D și conține două punți H complete, cu protecții incluse: la el, combinațiile „interzise” pe intrări nu produc scurtcircuit, ci doar opresc motorul.

SECȚIUNEA 02 · PIESA DE AZI

Piesă nouă: Circuitul integrat L293D

PRIMA APARIȚIE



CE ESTE?

Un circuit integrat cu 16 terminale (pini), care conține două punți H gata construite. Poate conduce până la 600 de miliamperi pe fiecare punte, de zeci de ori peste cât suportă un pin de placă. Punctul de pe capsulă indică de unde începe numerotarea: pe un rând se află prima punte, pe celălalt, a doua.

SECȚIUNEA 02 · PIESA DE AZI

Piesă nouă: Circuitul integrat L293D

PRIMA APARIȚIE

CE FACE?

Cele 16 terminale au doar cinci roluri, repetate. Input 1 și Input 2 primesc comanda plăcii, iar Output 1 și Output 2 livrează motorului curentul mare: Input 1 = 1 și Input 2 = 0 → înainte; invers → înapoi; ambele la fel → motorul stă. Enable 1 2 este comutatorul general al acestei punți: fără 1 pe el, ieșirile rămân inactive. Power 1 alimentează electronica cipului, iar Power 2 curentul motorului; sunt separate pentru ca, în realitate, motorul să poată avea baterie proprie. Cele patru Ground sunt conectate între ele în interior, deci este suficient să conectezi unul. Restul: Input 3, Input 4, Output 3, Output 4 și Enable 3 4 formează a doua punte, identică cu prima; astăzi rămâne nefolosită.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

În roboții de la concursurile școlare și în nenumărate jucării cu telecomandă: oriunde un motor trebuie să funcționeze în ambele sensuri.

DE CE NE TREBUIE AZI?

El transformă „înainte și înapoi” dintr-o operație cu șurubelnița într-o simplă pereche de instrucțiuni din program.

SECȚIUNEA 03 · PRACTICĂ

Înainte, pauză, înapoi: din program

Astăzi apare o componentă în plus: o baterie. Cipul are nevoie de cel puțin 4,5 V pentru partea sa de comandă, iar pe tranzistoarele sale interne mai cade o tensiune de aproximativ 2,5 V până la motor; tensiunea plăcii (3,3 V la micro:bit, 5 V la Arduino) este prea mică pentru a acoperi confortabil ambele nevoi. Bateria alimentează cipul și motorul; placa doar comandă.

Cipul are 16 terminale, dar astăzi folosești doar opt; restul aparțin celei de-a doua punți. Le conectezi pe roluri:

SECȚIUNEA 03 · PRACTICĂ

Alimentarea · Power 1, Power 2 și Enable 1&2 → plusul bateriei

Masa comună · minusul bateriei și GND al plăcii → Ground al cipului

Comanda · Input 1 și Input 2 → cei doi pini din antetul traseului tău

Forța · Output 1 și Output 2 → cele două borne ale motorului

Cele două alimentări separate despre care vorbea fișa cipului își arată acum rostul: bateria conduce curentul mare, iar placa rămâne protejată. Pentru ca cele două surse să lucreze împreună, minusurile lor trebuie conectate: acest punct comun se numește masă comună și reprezintă regula numărul unu ori de câte ori un montaj are două alimentări. Fără masă comună, cipul nu recunoaște comenzile plăcii.

Placa nu atinge niciodată motorul: îi transmite cipului comanda prin cele două intrări, iar cipul livrează curentul mare prin cele două ieșiri. Sensul este dat de perechea de valori: 1 și 0 înseamnă înainte, 0 și 1 înapoi, iar două valori egale înseamnă stop.

Programul de astăzi este o secvență temporizată, construită doar din ce știi deja: scrie pe pini, așteaptă, repetă, ca un semafor care își schimbă singur culorile.

SECȚIUNEA 03 · PRACTICĂ

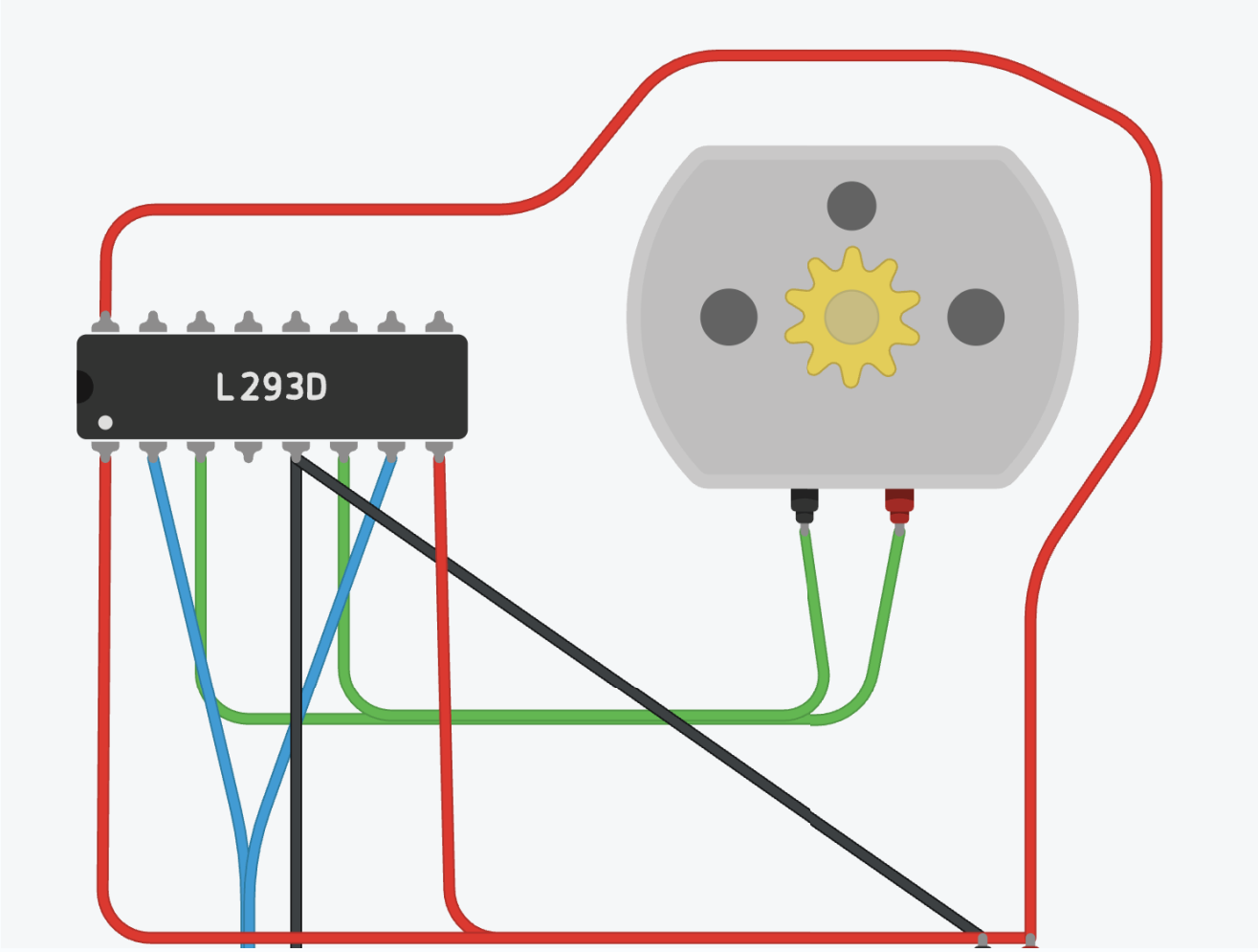


Fig. 1. Puntea gata construită: placa transmite comenzi prin Input 1 și Input 2; motorul primește curent prin Output 1 și Output 2.

LISTA DE CABLARE NOUĂ FIRE, BIFEAZĂ-LE PE RÂND

Traseul A · Micro:bit

ALIMENTARE · ROȘU
baterie + → L293D Power 1
baterie + → L293D Power 2
baterie + → L293D Enable 1&2

MASĂ COMUNĂ · NEGRU
baterie − → L293D Ground
GND → L293D Ground

COMANDĂ · ALBASTRU
P0 → L293D Input 1
P1 → L293D Input 2

FORȚĂ · VERDE
L293D Output 1 → motor, o bornă a bateriei
L293D Output 2 → motor, cealaltă bornă a bateriei

Traseul B · Arduino Uno

ALIMENTARE · ROȘU
baterie + → L293D Power 1
baterie + → L293D Power 2
baterie + → L293D Enable 1&2

MASĂ COMUNĂ · NEGRU
baterie − → L293D Ground
GND → L293D Ground

COMANDĂ · ALBASTRU
pinul 8 → L293D Input 1
pinul 9 → L293D Input 2

FORȚĂ · VERDE
L293D Output 1 → motor, o bornă
L293D Output 2 → motor, cealaltă bornă

SECȚIUNEA 03 · PRACTICĂ

Până acum forța venea direct din alimentare, așa că am folosit roșu pentru amândouă. Astăzi cipul le separă: roșu rămâne alimentarea care intră, verde este forța care iese spre motor.

Bateria este un suport de 4 baterii AA (6 V) și se încadrează exact în intervalul de 4,5-7 V cerut de cip. Cele patru terminale Ground sunt conectate între ele în interior, deci ambele fire negre pot merge la același terminal (în montajele reale se conectează toate patru, deoarece ajută și la disiparea căldurii). Ordinea firelor la motor nu contează: ea decide doar care sens îl numim „înainte”.

TRASEUL A · MICRO:BIT

Input 1 → P0 · Input 2 → P1 · alimentarea cipului și a motorului: baterie 4×AA

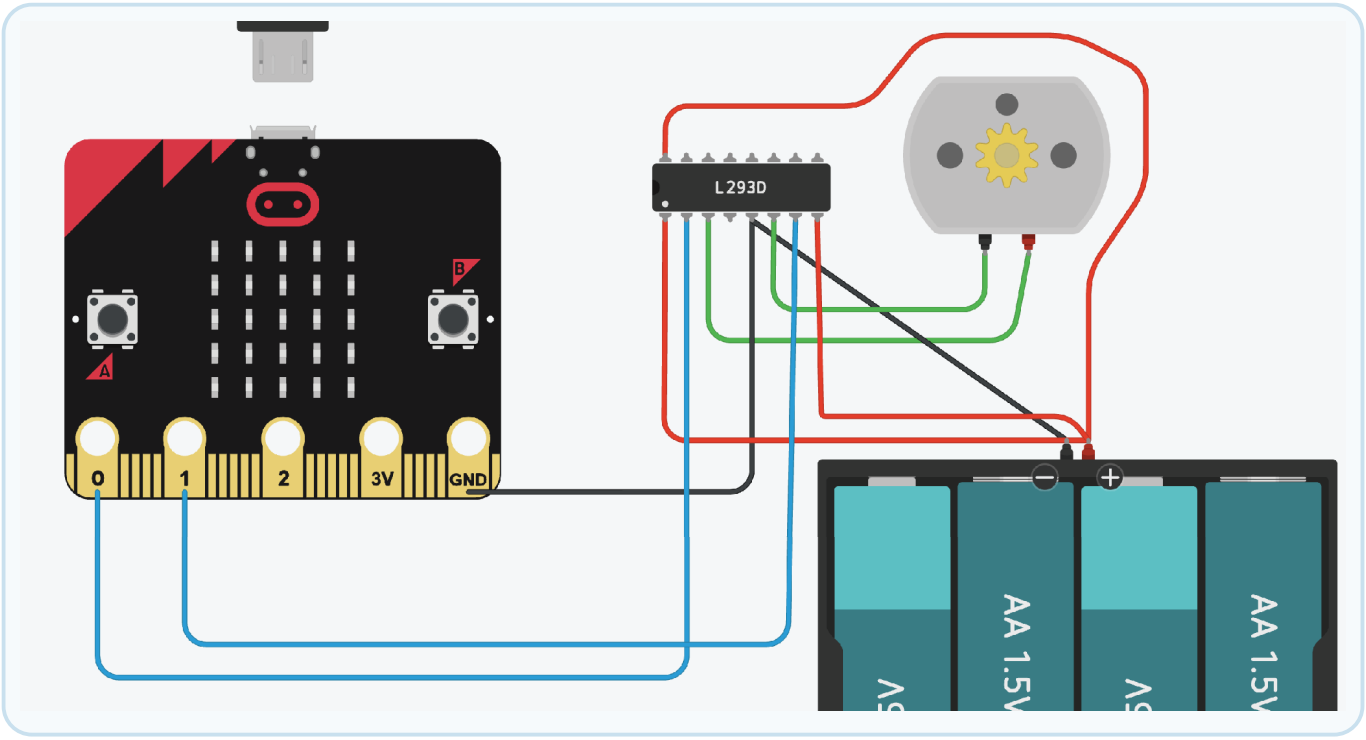
CIRCUIT DE START

▶ Deschide circuitul de start 2.4 · Micro:bit

Placa, bateria, cipul L293D și motorul se află pe planșa de lucru. Apasă „Copy and Tinker” și cablează după schemă.

1 Cablează puntea și verifică de două ori

Trasează firele exact ca în figura de mai jos. Astăzi ai multe fire: bifează-le pe schemă unul câte unul, cu creionul, pe măsură ce le trasezi. Compară apoi cu captura.



☐ **Toate firele sunt la locul lor**

TRASEUL A · MICRO:BIT
Input 1 → P0 · Input 2 → P1 · alimentarea cipului și a motorului: baterie 4×AA

2 Scrie secvența de sens

În blocuri: forever → digital write pin P0 to HIGH, digital write pin P1 to LOW, wait 2000 ms → digital write pin P0 to LOW, digital write pin P1 to LOW, wait 1000 ms → digital write pin P0 to LOW, digital write pin P1 to HIGH, wait 2000 ms → digital write pin P0 to LOW, digital write pin P1 to LOW, wait 1000 ms. La fiecare fază scrii amândoi pinii, chiar dacă unul rămâne neschimbat: perechea decide sensul, nu un singur pin. Rulează, apoi treci pe Python (mai jos). Privește direcția motorului: înainte... pauză... înapoi... pauză.

☐ **Motorul își schimbă singur sensul**

VARIANTA ÎN TEXT · ACELEAȘI INSTRUCȚIUNI **PYTHON · MICRO:BIT**

TRASEUL A · MICRO:BIT

Input 1 → P0 · Input 2 → P1 · alimentarea cipului și a motorului: baterie 4×AA

VARIANTA ÎN TEXT · ACELEAȘI INSTRUCȚIUNI — PYTHON · MICRO:BIT

```
# Lecția 2.4: înainte, pauză, înapoi
# (varianta Tinkercad / MakeCode)
def on_forever():   # blocul pe care placa îl repetă la nesfârșit
    # ---- faza 1: înainte, 2 secunde ----
    pins.digital_write_pin(DigitalPin.P0, 1)   # Input 1 = 1 → deschide o diagonală a punții
    pins.digital_write_pin(DigitalPin.P1, 0)   # Input 2 = 0 → cealaltă rămâne închisă
    basic.pause(2000)   # ține sensul 2000 ms = 2 secunde
    # ---- faza 2: pauză, 1 secundă ----
    pins.digital_write_pin(DigitalPin.P0, 0)   # ambele intrări egale...
    pins.digital_write_pin(DigitalPin.P1, 0)   # ...înseamnă motor oprit, nu scurtcircuit
    basic.pause(1000)
    # ---- faza 3: înapoi, 2 secunde ----
    pins.digital_write_pin(DigitalPin.P0, 0)   # Input 1 = 0
    pins.digital_write_pin(DigitalPin.P1, 1)   # Input 2 = 1 → se deschide diagonala opusă
    basic.pause(2000)   # aceeași durată, sens invers
    # ---- faza 4: pauză, apoi de la capăt ----
    pins.digital_write_pin(DigitalPin.P0, 0)
    pins.digital_write_pin(DigitalPin.P1, 0)
    basic.pause(1000)
basic.forever(on_forever)   # forever din blocuri = acest rând în Python
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Input 1 → pinul 8 · Input 2 → pinul 9 · alimentarea cipului și a motorului: baterie 4×AA

CIRCUIT DE START

▶ Deschide circuitul de start 2.4 · Arduino

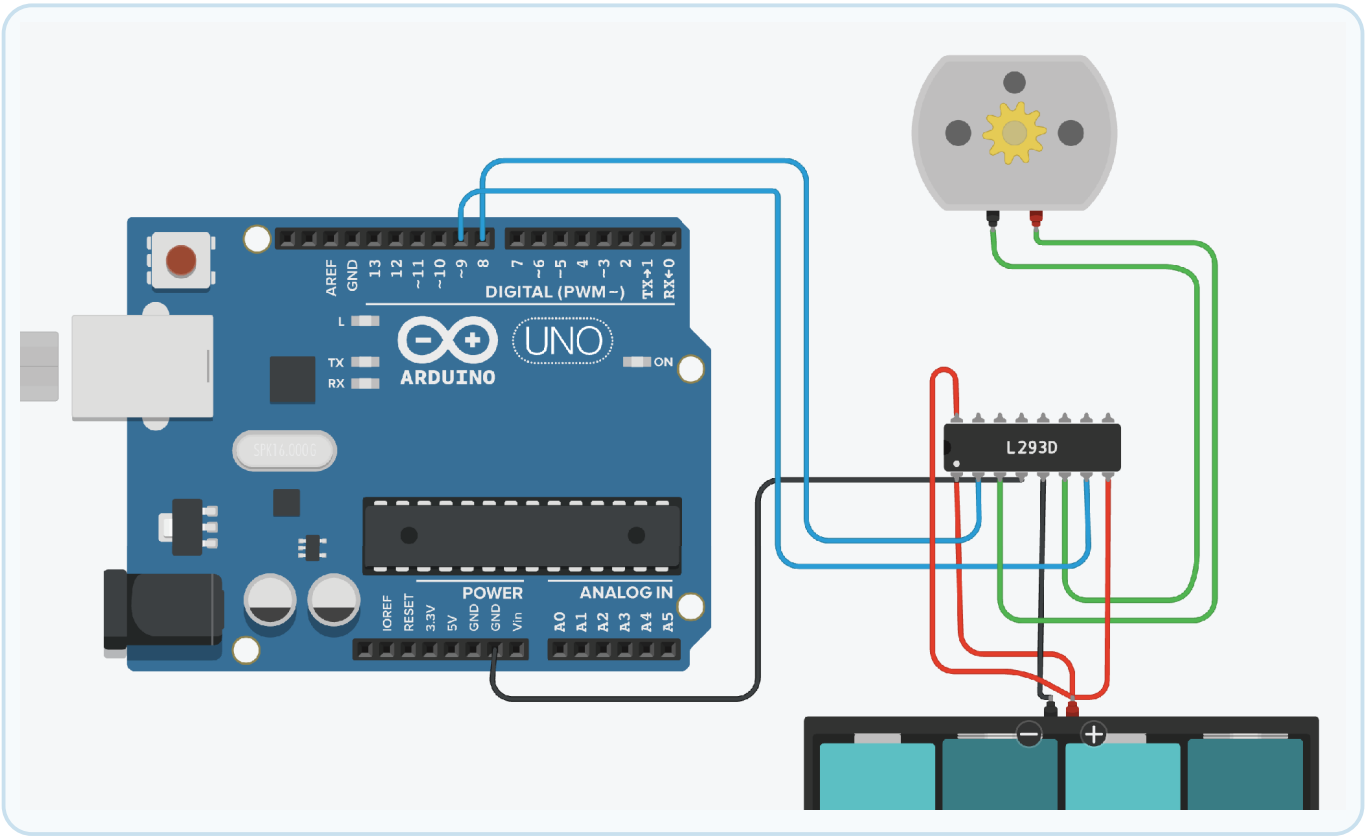
Aceleași componente, cu Arduino Uno. Apasă „Copy and Tinker” și cablează după aceeași schemă.

TRASEUL B · ARDUINO UNO

Input 1 → pinul 8 · Input 2 → pinul 9 · alimentarea cipului și a motorului: baterie 4×AA

1 Cablează și programează în blocuri

Aceeași schemă ca la Micro:bit, aceeași baterie, aceeași masă comună; se schimbă doar numele pinilor de comandă: 8 și 9 în loc de P0 și P1. În blocuri: forever → pin 8 HIGH, pin 9 LOW, wait 2000 ms → pin 8 LOW, pin 9 LOW, wait 1000 ms → pin 8 LOW, pin 9 HIGH, wait 2000 ms → pin 8 LOW, pin 9 LOW, wait 1000 ms. Ca la Micro:bit, scrii ambii pini la fiecare fază. Deschide „Blocks + Text” și privește vitrina: aceleași instrucțiuni, scrise cu digitalWrite și delay; le recunoști deja din Lecția 2.3.



TRASEUL B · ARDUINO UNO
Input 1 → pinul 8 · Input 2 → pinul 9 · alimentarea cipului și a motorului: baterie 4×AA

1 Cablează și programează în blocuri

Aceeași schemă ca la Micro:bit, aceeași baterie, aceeași masă comună; se schimbă doar numele pinilor de comandă: 8 și 9 în loc de P0 și P1. În blocuri: forever → pin 8 HIGH, pin 9 LOW, wait 2000 ms → pin 8 LOW, pin 9 LOW, wait 1000 ms → pin 8 LOW, pin 9 HIGH, wait 2000 ms → pin 8 LOW, pin 9 LOW, wait 1000 ms. Ca la Micro:bit, scrii ambii pini la fiecare fază. Deschide „Blocks + Text” și privește vitrina: aceleași instrucțiuni, scrise cu digitalWrite și delay; le recunoști deja din Lecția 2.3.

```cpp
// C++ code
//
void setup()
{
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
}

void loop()
{
  digitalWrite(8, HIGH);
  digitalWrite(9, LOW);
  delay(2000); // Wait for 2000 millisecond(s)
  digitalWrite(8, LOW);
  digitalWrite(9, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(8, LOW);
  digitalWrite(9, HIGH);
  delay(2000); // Wait for 2000 millisecond(s)
  digitalWrite(8, LOW);
  digitalWrite(9, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

☐ **Motorul își schimbă singur sensul**

C++ generat, cu comentarii în limba română **C++ · ARDUINO**

TRASEUL B · ARDUINO UNO

Input 1 → pinul 8 · Input 2 → pinul 9 · alimentarea cipului și a motorului: baterie 4×AA

C++ generat, cu comentarii în limba română — C++ · ARDUINO

```
// Lecția 2.4: Înainte, pauză, înapoi (varianta Arduino C++)
void setup()
{
  pinMode(8, OUTPUT); // Pin comutare direcție 1
  pinMode(9, OUTPUT); // Pin comutare direcție 2
}
void loop()
{
  // ---- Faza 1: Înainte, 2 secunde ----
  digitalWrite(8, HIGH); // Input 1 = 1
  digitalWrite(9, LOW);  // Input 2 = 0 -> merge înainte
  delay(2000);           // menține sensul 2 secunde
  // ---- Faza 2: Pauză, 1 secundă ----
  digitalWrite(8, LOW);  // ambele 0
  digitalWrite(9, LOW);  // -> motor oprit
  delay(1000);           // pauză 1 secundă
  // ---- Faza 3: Înapoi, 2 secunde ----
  digitalWrite(8, LOW);  // Input 1 = 0
  digitalWrite(9, HIGH); // Input 2 = 1 -> merge înapoi
  delay(2000);           // menține sensul invers 2 secunde
  // ---- Faza 4: Pauză, apoi reluare ----
  digitalWrite(8, LOW);  // ambele 0
  digitalWrite(9, LOW);  // -> motor oprit
  delay(1000);           // pauză 1 secundă
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT

Are exact trei pini mari (P0, P1, P2); astăzi doi sunt deja ocupați de punte. Se simte că pinii sunt puțini.

ARDUINO UNO

Are 14 pini digitali; puntea ocupă doi și rămân suficienți pentru tot ce urmează. La roboții cu multe motoare, acest lucru contează.

Concluzia: cablajul și logica sunt identice; diferă doar câți pini îți rămân liberi. Primul criteriu real de alegere între plăci: numărul de pini.

SECȚIUNEA 04 · DEPANARE

⚠ Motorul stă, deși codul rulează?

Verifică în ordine: 1) Bateria este conectată la Power 1, Power 2 și Enable? Fără Enable, puntea rămâne inactivă. 2) Minusul bateriei ajunge la același Ground ca GND-ul plăcii? Fără masă comună, cipul nu recunoaște comenzile. 3) Intrările nu sunt cumva amândouă la aceeași valoare? (egale = stop) 4) Merge doar într-un sens → a doua intrare este neconectată sau pe alt pin decât în cod. 5) Motorul se mișcă slab și neregulat → ai lăsat alimentarea cipului pe placă, nu pe baterie.

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă? (8 minute, cu mișcare)

Pe podea, cu bandă de hârtie, se desenează un H mare. Patru elevi sunt comutatoarele S1-S4 (câte unul pe fiecare colț), iar un elev, în bara din mijloc, este motorul: ține o săgeată de carton. Clasa strigă o combinație, de exemplu „S1 și S4!”: comutatoarele strigate se așază în genunchi (închis), iar motorul rotește săgeata în sensul corect. „S2 și S3!” → sens invers. Dacă cineva strigă o combinație de pe aceeași parte, toată clasa strigă „SCURTCIRCUIT!” și cei doi vinovați ies o tură. Ultimele runde: viteză maximă, fără timp de gândire.

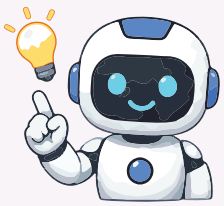
Varianta Kahoot: scheme de punte cu combinații date: „înainte, înapoi, stop sau scurtcircuit?”;linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 2.4-J: tabelul combinațiilor; elevii completează efectul fiecăreia dintre cele 6 combinații posibile

Varianta fără ecrane: chiar jocul principal; necesită bandă de hârtie, o săgeată de carton și spațiu între bănci

Provocare: pentru cei rapizi

Manevra de parcare. Scrie secvența pentru o mașinuță care parchează: înainte 1 secundă, pauză scurtă, înapoi o jumătate de secundă, stop definitiv. Atenție la ultima cerință: cum faci ca programul să NU se repete la nesfârșit? Indiciu: ce se întâmplă dacă instrucțiunile nu stau în forever?



3 întrebări rapide

1. Ce pereche de comutatoare închizi pentru „înainte” și ce pereche pentru „înapoi”?
2. De ce este interzisă închiderea a două comutatoare de pe aceeași parte?
3. Ce rol are pinul Enable al circuitului L293D?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

punte H: patru comutatoare așezate în H, care inversează polaritatea motorului

scurtcircuit: traseu direct de la alimentare la GND, ocolind consumatorul

circuit integrat (cip): mai multe circuite complete, închise într-o capsulă cu terminale (pini)

Input / Output: intrare (comanda primită) / ieșire (curentul livrat); denumirile englezești de pe cip

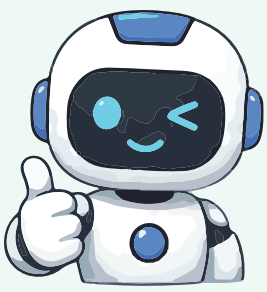
Enable: comutatorul general al punții: fără 1 aici, motorul nu primește nimic

masă comună: minusurile a două alimentări diferite, conectate împreună; fără ea, cele două surse nu pot lucra împreună

secvență: listă de instrucțiuni executate pe rând, cu pauze; programul de astăzi

Ce trebuie să știi după această lecție?

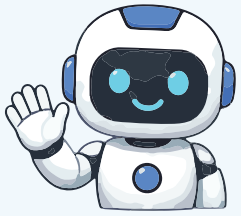
- ✓ Puntea H inversează polaritatea motorului din comutatoare, nu din fire mutate cu mâna
- ✓ Se închid doar diagonale opuse; aceeași parte = scurtcircuit
- ✓ L293D = două punți gata construite, cu protecții; intrări egale înseamnă stop, nu pericol
- ✓ Sensul motorului = o pereche de valori pe Input 1 și Input 2, scrise din program
- ✓ Motorul are alimentare proprie; placa doar comandă, iar minusurile se conectează împreună



LECȚIA 2.5

Controlul vitezei: semnalul PWM

Pinul digital cunoaște doar două stări: pornit și oprit. Și totuși, în această lecție motorul tău va funcționa cu viteză mică, medie și mare. Secretul este o tehnică folosită de toată electronica modernă.



CE VEI ȘTI LA FINALUL LECȚIEI?

- Să explici ce este semnalul PWM și ce măsoară factorul de umplere
- Să calculezi valoarea de scris pe pin pentru un procent dorit, pe fiecare placă
- Să comanzi motorul în trei trepte de viteză și să verifici efectul citind turația

SECȚIUNEA 01 · TEORIE

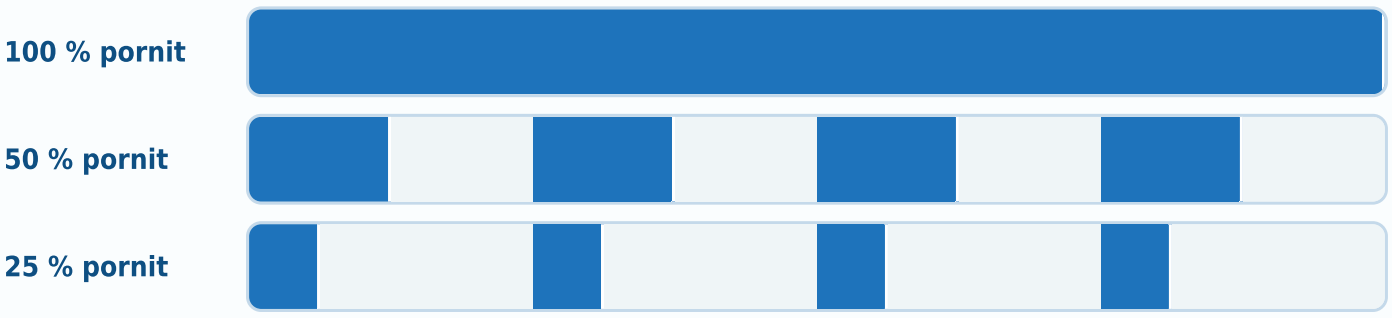
Pornit-oprit foarte repede: PWM

DEJA CUNOȘTI (DIN LECȚIILE 1.6 ȘI 2.3)

Blocul de scriere digitală pune pe pin doar HIGH sau LOW: totul sau nimic. Circuitul pin → tranzistor → motor din Lecția 2.3 rămâne astăzi neschimbat; schimbăm doar programul.

Cum obții „jumătate de viteză” dintr-un pin care cunoaște doar 1 și 0? Nu cobori tensiunea, ci pornești și oprești pinul de sute de ori pe secundă. Motorul, datorită inerției sale, nu apucă să se oprească între impulsuri: el resimte doar valoarea medie. Pinul pornit 50% din timp → motorul funcționează la aproximativ jumătate din viteză. Tehnica se numește PWM (din limba engleză: Pulse Width Modulation, adică modularea lățimii impulsurilor) și se află peste tot în jurul tău: variatorul care reduce lumina becului, treptele ventilatorului, luminozitatea ecranului pe care citești acum.

PWM văzut ca impulsuri



Aceeași amplitudine, alt timp petrecut în starea „pornit”.

SECȚIUNEA 01 · TEORIE

Formula lecției: factorul de umplere

factor de umplere = timpul cât pinul e pornit ÷ durata unei perioade × 100 %

O perioadă = un ciclu complet pornit + oprit, repetat de sute sau mii de ori pe secundă. Pornit tot timpul = 100 % = viteză maximă. Pornit jumătate din timp = 50 %. Mereu oprit = 0 %. Atât: restul lecției este doar traducerea procentului în numărul pe care îl înțelege placa ta.

Comanda se numește scriere analogică: în loc să alegi între HIGH și LOW, scrii un număr pe o scară. Denumirea este însă o convenție: pinul nu produce o tensiune intermediară continuă, ci impulsuri PWM a căror valoare medie corespunde numărului scris. Aici apare prima diferență reală dintre plăci: Micro:bit numără de la 0 la 1023, iar Arduino de la 0 la 255. Procentul dorit îl transformi în valoare printr-o simplă proporție: 50 % din 1023 ≈ 512; 50 % din 255 ≈ 128.

SECȚIUNEA 02 · EXERSĂM

Două conversii pe caiet, înainte de cod. Scrie proporția completă.

Exercițiul 1 · Vrei 75 % pe Arduino. Ce valoare scrii? (75 ÷ 100 × 255 = ?)

Exercițiul 2 · Vrei 10 % pe Micro:bit. Ce valoare scrii? Merită să pornească un motor de la ea? (Verifici imediat în simulator.)

SECȚIUNEA 03 · PRACTICĂ

Trei trepte de viteză, cu dovada în turație

Circuitul rămâne cel din Lecția 2.3. Astăzi schimbi doar programul și demonstrezi cu cifre că funcționează: trei trepte de viteză, trei turații citite.

TRASEUL A · MICRO:BIT
Scara: 0 – 1023 · oricare pin mare (P0, P1, P2) acceptă scriere analogică

CHECKPOINT (DACĂ NU MAI AI CIRCUITUL DIN 2.3)
▶ Circuit Micro:bit gata cablat: pin → tranzistor → motor
Placa, motorul, tranzistorul, rezistorul și dioda se află gata cablate pe planșă. Apasă „Copy and Tinker" și treci direct la cod.

TRASEUL A · MICRO:BIT
Scara: 0 – 1023 · oricare pin mare (P0, P1, P2) acceptă scriere analogică

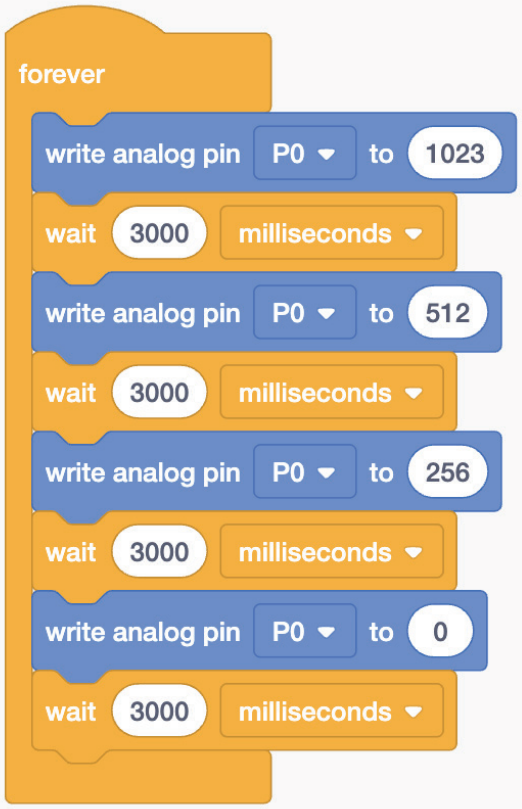
1 Deschide circuitul și pregătește tabelul

Nu schimbi niciun fir: este exact montajul din Lecția 2.3 (pinul P0 → rezistor → bază, motorul pe colector, dioda în paralel cu motorul). Deschide-l din contul tău sau folosește checkpoint-ul de mai sus. Desenează apoi în caiet tabelul cu trei coloane: treapta · valoarea scrisă · turația citită. Îl completezi în timp ce programul rulează.

☐ **Circuitul este deschis și tabelul este desenat în caiet**

2 Construiește treptele în blocuri

Apasă „Code". În Blocuri: forever → write analog pin P0 to 1023, wait 3000 ms → write analog pin P0 to 512, wait 3000 ms → write analog pin P0 to 256, wait 3000 ms → write analog pin P0 to 0, wait 3000 ms. Blocul de scriere analogică are o casetă de număr, nu un meniu HIGH/LOW; acolo introduci valoarea. Rulează, execută clic pe motor la fiecare treaptă și notează turația în tabel. Comută apoi pe Python: același program, rând cu rând.



☐ **Am completat tabelul cu turațiile celor trei trepte**

TRASEUL A · MICRO:BIT

Scara: 0 – 1023 · oricare pin mare (P0, P1, P2) acceptă scriere analogică

VARIANTA ÎN TEXT · ACELEAȘI INSTRUCȚIUNI

PYTHON · MICRO:BIT

```
# Lecția 2.5: trei trepte de viteză
# (varianta Tinkercad / MakeCode)
def on_forever():   # blocul pe care placa îl repetă la nesfârșit
    # analog_write_pin, nu digital: aici scrii un număr de pe scara 0–1023,
    # nu doar 1 sau 0. Numărul este factorul de umplere, tradus în valoare.
    pins.analog_write_pin(AnalogPin.P0, 1023)   # 1023 din 1023 = 100 % → viteză maximă
    basic.pause(3000)   # ține treapta 3 secunde: ai timp să citești turația
    pins.analog_write_pin(AnalogPin.P0, 512)   # 512 ≈ jumătate din scară = 50 %
    basic.pause(3000)
    pins.analog_write_pin(AnalogPin.P0, 256)   # 256 ≈ un sfert din scară = 25 %
    basic.pause(3000)   # aici motorul poate nici să nu mai pornească; notează ce observi
    pins.analog_write_pin(AnalogPin.P0, 0)   # 0 = pin oprit tot timpul → motorul se oprește
    basic.pause(3000)   # pauză, apoi de la capăt
basic.forever(on_forever)   # forever din blocuri = acest rând în Python
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Scara: 0 – 255 · doar pinii marcați cu ~ (3, 5, 6, 9, 10, 11) știu PWM

CHECKPOINT (DACĂ NU MAI AI CIRCUITUL DIN 2.3)

▶ Circuit Arduino gata cablat: pin → tranzistor → motor

Placa, motorul, tranzistorul, rezistorul și dioda se află gata cablate pe planșă. Apasă „Copy and Tinker" și treci direct la cod.

1 Deschide circuitul și verifică pinul

Același montaj din Lecția 2.3, neschimbat. Un singur lucru contează astăzi: firul de comandă trebuie să fie conectat la un pin marcat cu ~. Circuitul din 2.3 folosea deja pinul 9, care are tildă, deci totul este în regulă. Desenează în caiet același tabel cu trei coloane.

☐ **Circuitul este deschis și tabelul este desenat în caiet**

TRASEUL B · ARDUINO UNO
Scara: 0 – 255 · doar pinii marcați cu ~ (3, 5, 6, 9, 10, 11) știu PWM

2 Blocuri și vitrina analogWrite

Apasă „Code" și alege modul „Blocks + Text". Construiește: forever → set pin 9 to 255, wait 3000 ms → set pin 9 to 128, wait 3000 ms → set pin 9 to 64, wait 3000 ms → set pin 9 to 0, wait 3000 ms. În dreapta, Tinkercad scrie analogWrite(9, 128) în loc de digitalWrite: „scrie analogic", exact ca scrierea analogică de pe micro:bit, doar cu altă scară. Rulează și completează tabelul.

☐ **Am completat tabelul cu turațiile celor trei trepte**

TRASEUL B · ARDUINO UNO

Scara: 0 – 255 · doar pinii marcați cu ~ (3, 5, 6, 9, 10, 11) știu PWM

C++ GENERAT, CU COMENTARII ÎN LIMBA ROMÂNĂ

C++ · ARDUINO

```
// C++ code
// Lecția 2.5: trei trepte de viteză (varianta Arduino)
void setup() {   // se execută o singură dată, la pornire
  pinMode(9, OUTPUT);   // pinul 9 devine pin de ieșire
}
void loop() {   // se repetă la nesfârșit
  // analogWrite, nu digitalWrite: al doilea număr este valoarea de pe scara 0–255
  analogWrite(9, 255);   // 255 din 255 = 100 % → viteză maximă
  delay(3000);   // ține treapta 3 secunde: ai timp să citești turația
  analogWrite(9, 128);   // 128 ≈ jumătate din scară = 50 %
  delay(3000);
  analogWrite(9, 64);   // 64 ≈ un sfert din scară = 25 %
  delay(3000);   // aici motorul poate nici să nu mai pornească; notează ce observi
  analogWrite(9, 0);   // 0 = pin oprit tot timpul → motorul se oprește
  delay(3000);   // pauză, apoi de la capăt
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT

Scara 0–1023: pași mai fini (de 4 ori mai multe trepte posibile). Scriere analogică pe oricare dintre pinii mari.

ARDUINO UNO

Scara 0–255 și PWM doar pe pinii cu ~. Mai puține trepte, dar 6 pini PWM: poți comanda mai multe motoare deodată.

Concluzia: aceeași idee, două scări. Când citești un proiect de pe internet, prima întrebare este mereu: pe ce scară lucrează placa lui?

SECȚIUNEA 04 · DEPANARE

⚠ Viteza nu se schimbă?

1) La Arduino: pinul tău are ~? Fără ~, analogWrite nu produce PWM adevărat. 2) Valorile sunt pe scara plăcii tale? 512 pe Arduino nu înseamnă 50 %, ci o valoare peste maximum. 3) La valori mici (sub aproximativ 15 %), motorul poate să nu pornească deloc: nu este un defect, ci fizică; pentru a se urni, motorul are nevoie de un cuplu minim, iar frecarea și inerția se opun pornirii.

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă? (6 minute, pe echipe)

Profesorul (sau un metronom de telefon) bate ritmul: 10 bătăi = o perioadă. Echipa la rând primește în secret un factor de umplere, de exemplu 70 %, și ține un cartonaș verde ridicat exact 7 bătăi din 10, apoi coborât 3, în buclă, două perioade la rând. Celelalte echipe numără și strigă procentul. Punct pentru ghicit corect, punct pentru executat corect. Runda finală, invers: profesorul strigă „30 %!”, iar toate echipele trebuie să îl execute simultan, perfect sincron.

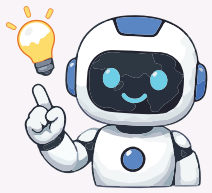
Varianta Kahoot: conversii procent ↔ valoare pe ambele scări, contra cronometru; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 2.5-J: drept-unghiuri-perioadă de colorat pentru 25 %, 50 %, 75 %, plus conversiile pe cele două scări

Varianta fără ecrane: chiar jocul principal; necesită doar un ritm bătut din palme și cartonașe

Provocare: pentru cei rapizi

Pornirea lină. Mașinile electrice nu pleacă smucit: viteza crește treptat. Scrie o secvență cu cinci trepte crescătoare (20 % → 40 % → 60 % → 80 % → 100 %), cu o jumătate de secundă între ele. Calculează mai întâi cele cinci valori pentru placa ta.



3 întrebări rapide

1. Ce este factorul de umplere și cum îl calculezi?
2. Ce valoare scrii pentru 25 % pe Micro:bit? Dar pe Arduino?
3. De ce nu pornește motorul la un factor de umplere foarte mic, deși LED-ul ar lumina?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

PWM: comandă prin impulsuri rapide pornit-oprit; consumatorul resimte doar valoarea medie

impuls: o porțiune scurtă de „pornit” din semnal

perioadă: un ciclu complet pornit + oprit, care se repetă

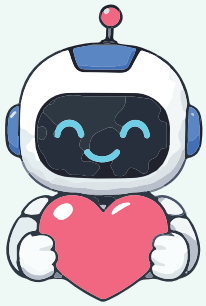
factor de umplere: procentul din perioadă în care pinul este pornit

scriere analogică: scrierea unei valori pe o scară (0–1023 sau 0–255), nu doar 0 și 1

pin PWM (~): pinii Arduino marcați cu ~, singurii care pot genera PWM

Ce trebuie să știi după această lecție?

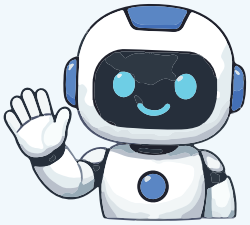
- ✓ PWM = pornit-oprit foarte repede; viteza urmează factorul de umplere
- ✓ Factor de umplere = timp pornit ÷ perioadă × 100 %
- ✓ Scările diferă: micro:bit 0–1023, Arduino 0–255 (doar pinii cu ~); procentul se traduce prin proporție
- ✓ Verificarea se face cu cifre: turația citită la fiecare treaptă, nu doar din ochi



De reținut vizual

LECȚIA 2.6 · 50 DE MINUTE

Servomotorul și primul mecanism: bariera de parcare



Ultimul motor din colecție este cel mai disciplinat: îi ceri un unghi și îl păstrează. Cu el construiești primul tău mecanism adevărat și închei modulul.

CE VEI ȘTI LA FINALUL LECȚIEI?

- Să explici prin ce se deosebește servomotorul de motoarele de până acum
- Să explici cum îi transmite placa unghiul dorit, folosind impulsurile PWM
- Să construiești și să programezi bariera de parcare care se ridică și coboară singură

SECȚIUNEA 01 · TEORIE

Motorul care ține poziția

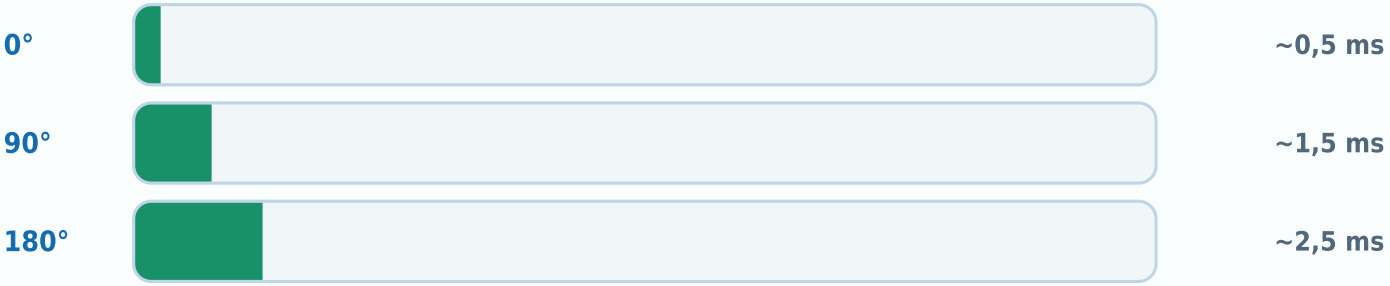
DEJA CUNOȘTI (DIN LECȚIILE 2.2 ȘI 2.5)

Reductorul schimbă viteza pe forță, iar PWM înseamnă impulsuri repetate, cu lățime controlată de tine. Astăzi cele două idei se întâlnesc în aceeași carcasă.

Motorul DC se rotește liber, cât timp are curent. Servomotorul este diferit: îi ceri un unghi între 0° și 180°, iar el se deplasează acolo și menține poziția, opunându-se dacă îl împingi. În carcasa lui se află trei elemente: un motor DC mic, un reductor (de aceea are forță) și un circuit de control care își măsoară singur unghiul și îl corectează permanent, în buclă închisă; de aceea ascultă.

Cum îi transmiți unghiul? Prin impulsuri: PWM-ul din lecția trecută, folosit de această dată ca mesaj, nu ca viteză. Servomotorul așteaptă câte un impuls la fiecare 20 de milisecunde (adică o frecvență de 50 Hz), iar lățimea impulsului codifică unghiul: un impuls de circa 0,5 ms înseamnă 0°, unul de 1,5 ms înseamnă 90°, iar unul de 2,5 ms înseamnă 180°. (Sunt valorile cu care lucrează Tinkercad; servomotoarele reale funcționează în jurul acelorași valori, cu mici diferențe de la model la model.) Îl întâlnești oriunde trebuie menținută o poziție: direcția mașinuțelor cu telecomandă, brațele robotice mici, clapetele de ventilație ale automobilelor.

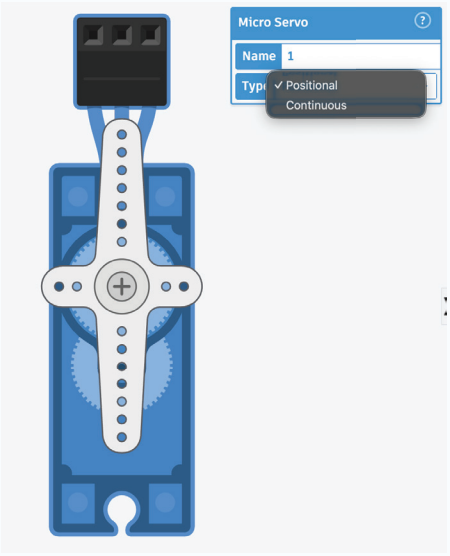
PWM folosit ca mesaj de poziție



Impulsurile se repetă la fiecare 20 ms (50 Hz), exact ca în textul lecției.

Piesă nouă: Micro-servomotorul

PRIMA APARIȚIE



CE ESTE?

O carcasă compactă cu un braț rotitor deasupra, numit corn, și trei fire: roșu (alimentare), maro sau negru (GND), portocaliu (semnalul de comandă). În Tinkercad: „Micro Servo”.

CE FACE?

Primește impulsuri pe firul de semnal și își poziționează cornul la unghiul cerut, între 0° și 180°, menținându-l acolo cu forța reductorului din interior. Poate fi setat în două moduri diferite: *Positional* (se deplasează la o poziție programată) sau *Continuous* (se rotește într-o direcție programată).

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ?

Mașinuțe și avioane cu telecomandă, brațe robotice de la concursuri, încuietori inteligente, clapetele de aer ale bordului de mașină.

DE CE NE TREBUIE AZI?

Este singurul motor care înțelege comanda „du-te la 90° și rămâi acolo”: exact ce îi trebuie unei bariere.

Teorie-vitrină: motorul pas cu pas

Mai există un motor în familie, pe care îl vei revedea în Modulul 4: motorul pas cu pas. El se rotește în pași ficși și egali (tipic 200 de pași pe rotație, adică exact 1,8° pasul), poate menține orice poziție și se poate roti continuu, cu precizie foarte mare. El mișcă masa și capul imprimantelor 3D. În Tinkercad nu există ca piesă de simulare (există doar biblioteca lui de cod, la Arduino), așa că aici rămâne la teorie; dar când imprimanta din Modulul 4 va desena straturi perfecte, vei ști cine trage de fire.

SECȚIUNEA 02 · PRACTICĂ · MINI-PROIECTUL MODULULUI

Bariera de parcare

Mecanismul: pe cornul servomotorului prinzi o barieră (în simulator, cornul însuși joacă rolul barierei; în varianta reală ar fi o fâșie de carton). Programul de astăzi o ridică la 90°, o menține 3 secunde, o coboară la 0°, așteaptă și reia ciclul: exact demonstrația unei bariere adevărate. Cablajul, comun: firul roșu → alimentare (3V / 5V), firul negru → GND, firul portocaliu → pinul de semnal. Semnalul este o comandă de mică putere; servomotorul își ia forța din firul roșu, nu din pin.

0°	90°	3 s
bariera coborâtă	bariera ridicată	poziția este menținută

SECȚIUNEA 02 · PRACTICĂ · MINI-PROIECTUL MODULULUI

Bariera de parcare

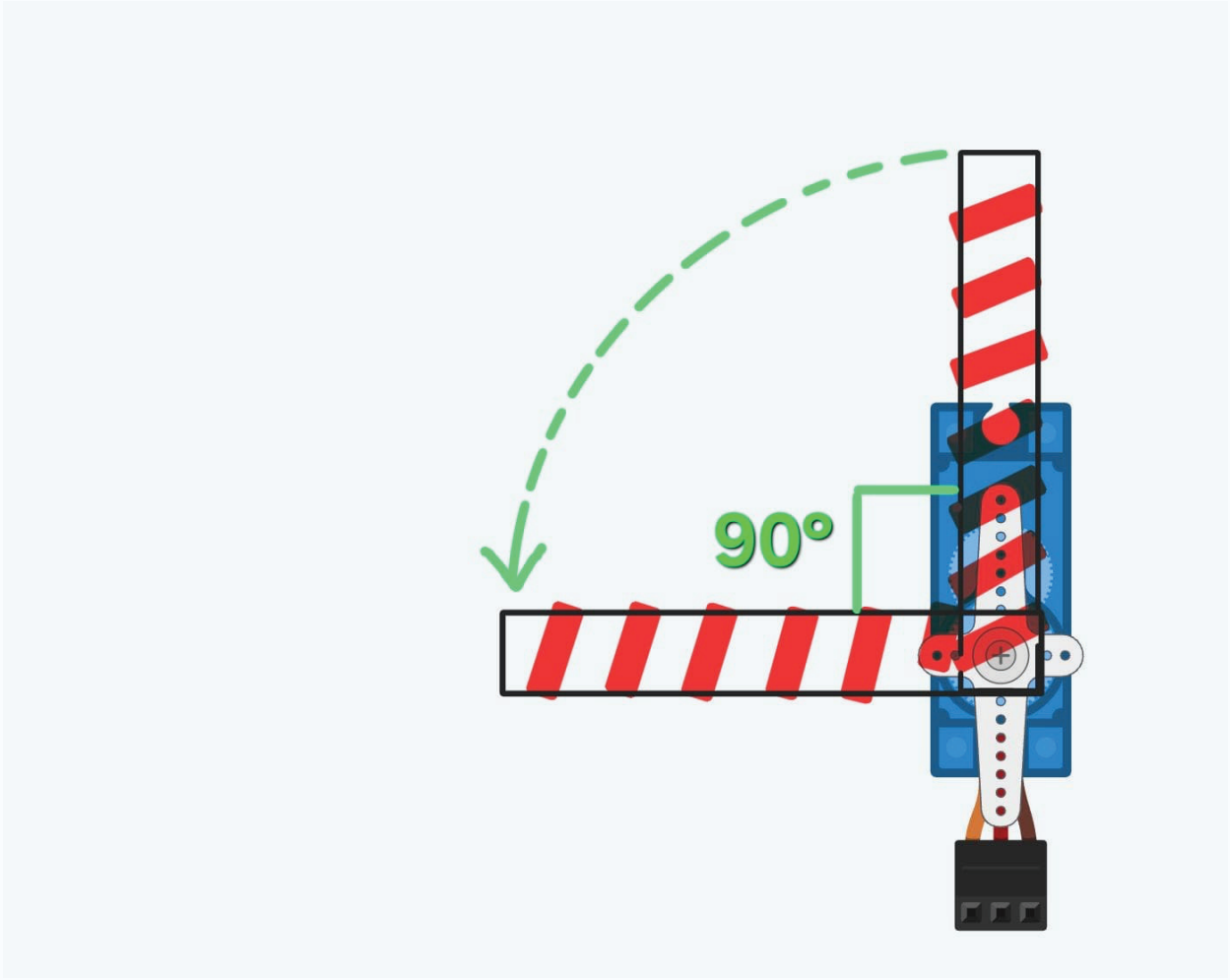


Fig. 1. Două comenzi, un mecanism: 0° jos, 90° sus.

TRASEUL A · MICRO:BIT

Semnalul → P0 · unghiul se traduce în valori de impuls, din Python

CIRCUIT DE START

▶ Deschide circuitul de start 2.6 · micro:bit

Placa și servomotorul, gata așezate. Apasă „Copy and Tinker”, cablează cele trei fire și treci la cod.

De reținut în cablaj

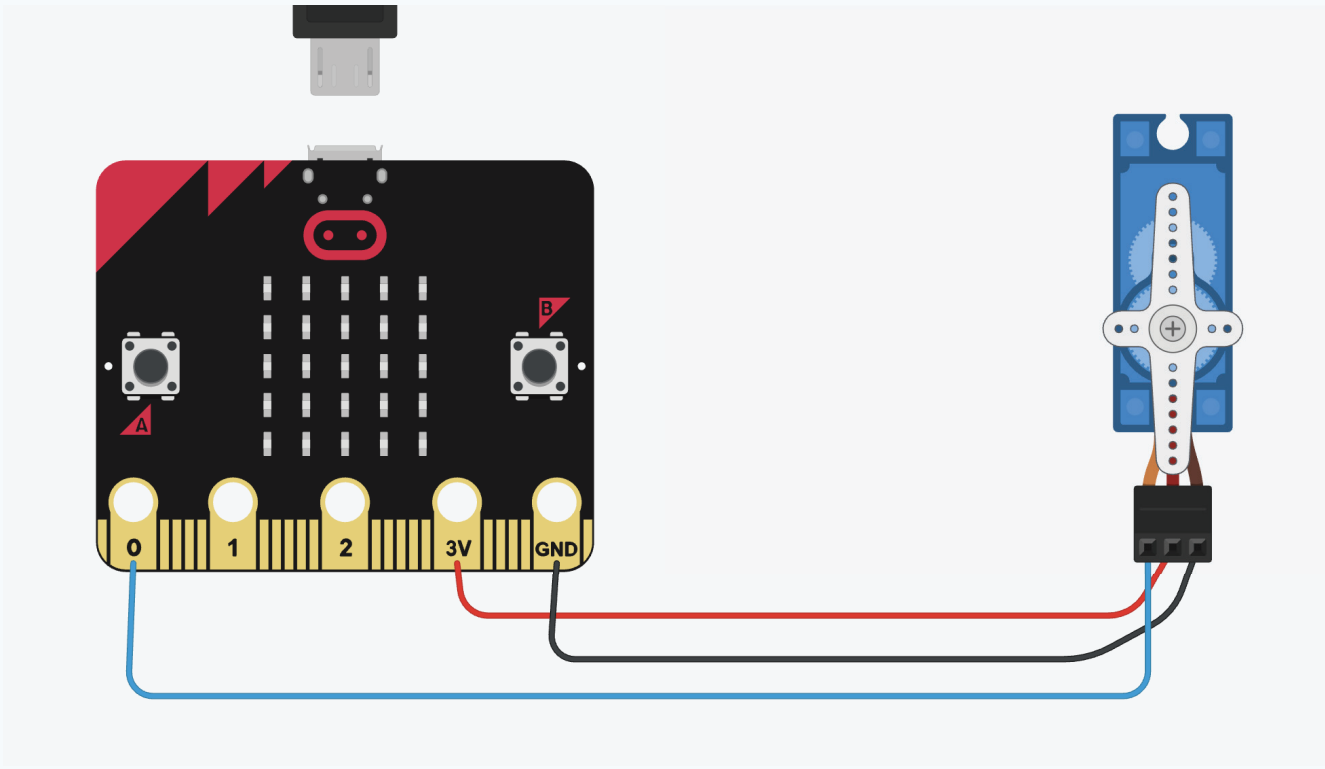
Semnalul este o comandă de mică putere; servomotorul își ia forța din firul roșu, nu din pin.

TRASEUL A · MICRO:BIT

Semnalul → P0 · unghiul se traduce în valori de impuls, din Python

1 Cablează cele trei fire

Servomotorul are trei fire, iar ordinea lor contează: roșu → 3V (de aici își ia forța), maro sau negru → GND, portocaliu → P0 (pe aici primește comanda). Firul roșu nu se conectează niciodată la un pin de comandă: pinul furnizează un semnal, nu curent de lucru.



☐ **Cele trei fire sunt la locul lor**

2 Setează limba servomotorului, apoi scrie ciclul

Placa Micro:bit nu are un bloc de servomotor, așa că astăzi lucrezi direct în Python: apasă „Code" și comută pe Python. Îi ceri mai întâi pinului să trimită un impuls la fiecare 20 ms, apoi scrii valori pe scara 0–1023 din Lecția 2.5. Este aceeași proporție, cu altă întrebare: ce parte din cele 20 ms trebuie să dureze impulsul? Pentru 90°: 1,5 ÷ 20 = 7,5 %, iar 7,5 % din 1023 ≈ 77. Tabelul complet: 0° ≈ 26 · 90° ≈ 77 · 180° ≈ 128. Dacă bariera nu ajunge exact la capăt, ajustează valoarea cu ±5; servomotoarele reale au și ele mici toleranțe.

0° ≈ 26	90° ≈ 77	180° ≈ 128
impuls ~0,5 ms	impuls ~1,5 ms	impuls ~2,5 ms

☐ **Bariera urcă și coboară singură**

TRASEUL A · MICRO:BIT

Semnalul → P0 · unghiul se traduce în valori de impuls, din Python

VARIANTA ÎN TEXT

PYTHON · MICRO:BIT

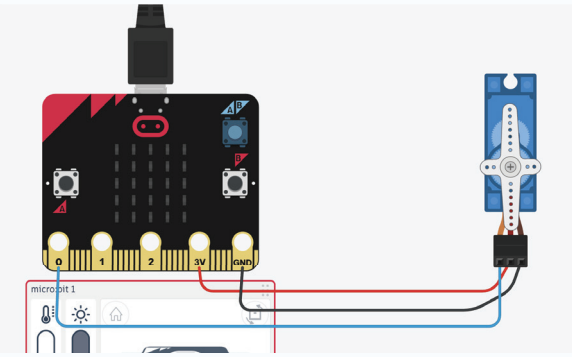
```
# Lecția 2.6: bariera de parcare
# (varianta Tinkercad / MakeCode)
# Perioada se setează o singură dată, la început: îi cere pinului P0
# să trimită câte un impuls la fiecare 20 ms, ritmul pe care îl așteaptă
# orice servomotor. Fără el, cornul tremură sau nu se mișcă deloc.
pins.analog_set_period(AnalogPin.P0, 20000)   # 20000 microsecunde = 20 ms
def on_forever():   # blocul pe care placa îl repetă la nesfârșit
    pins.analog_write_pin(AnalogPin.P0, 77)   # impuls de ~1,5 ms → 90° → bariera sus
    basic.pause(3000)   # ține bariera ridicată 3 secunde
    pins.analog_write_pin(AnalogPin.P0, 26)   # impuls de ~0,5 ms → 0° → bariera jos
    basic.pause(3000)   # pauză, apoi de la capăt
basic.forever(on_forever)   # forever din blocuri = acest rând în Python
```

COD GATA DE COPIAT

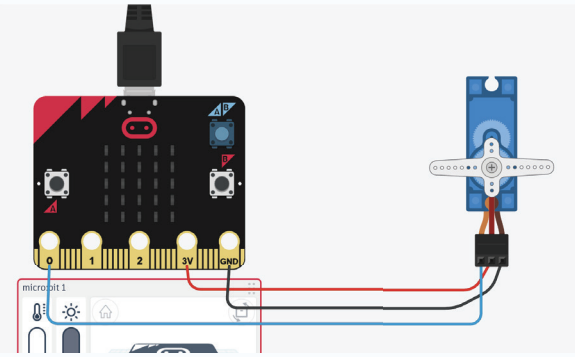
▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

Demonstrație:



Bariera la 0°



Bariera la 90°

TRASEUL B · ARDUINO UNO

Semnalul → pinul 9 · unghiul se cere direct în grade, din blocuri

La Arduino, unghiul se cere direct în grade, din blocuri.

TRASEUL B · ARDUINO UNO

Semnalul → pinul 9 · unghiul se cere direct în grade, din blocuri

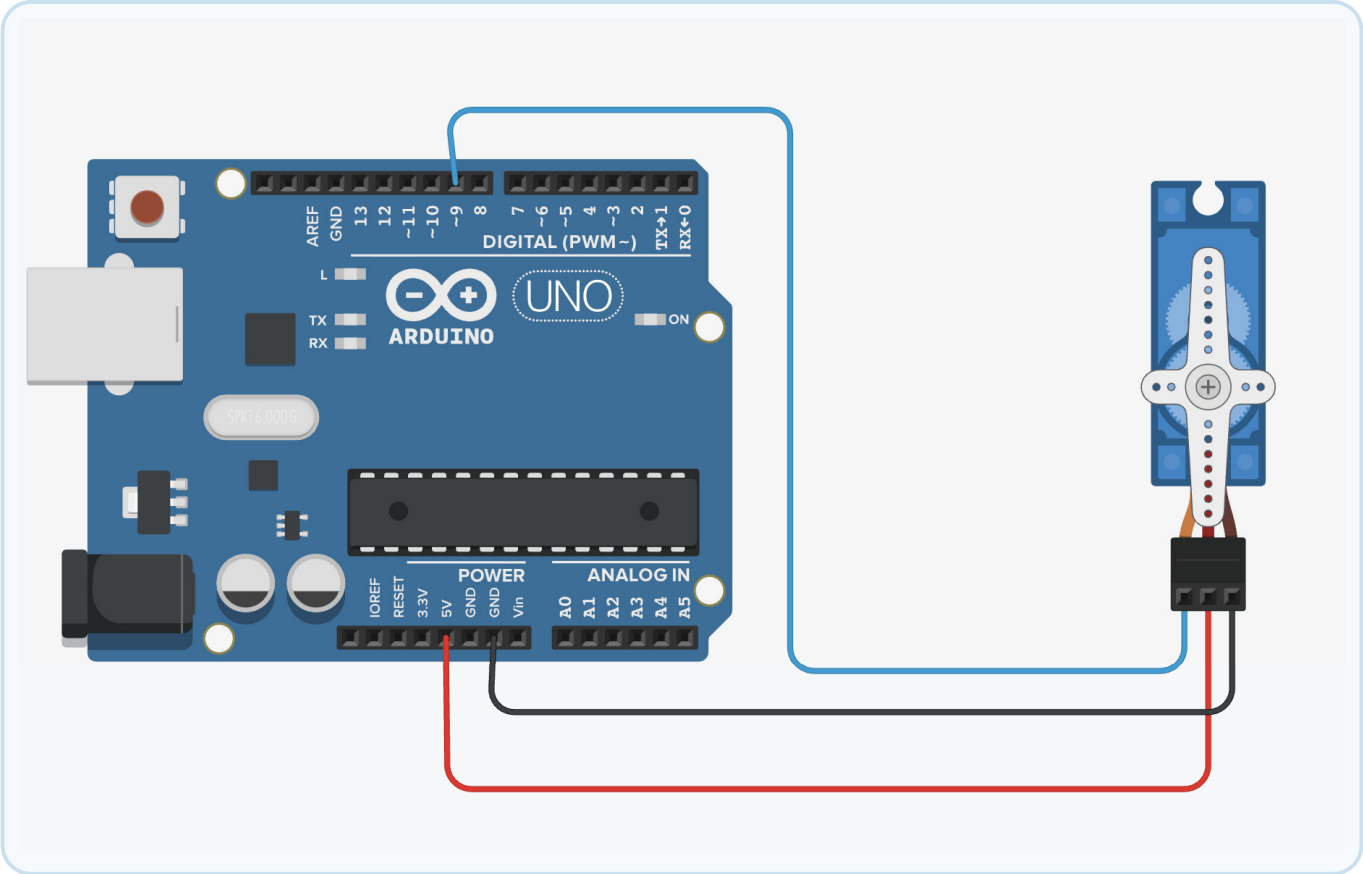
CIRCUIT DE START

▶ Deschide circuitul de start 2.6 · Arduino

Servomotorul și placa Arduino. Apasă „Copy and Tinker”, cablează cele trei fire și treci la cod.

1 Cablează aceleași trei fire

Același cablaj, cu alte nume: roșu → 5V, maro sau negru → GND, portocaliu → pinul 9. Servomotorul acceptă orice pin digital, dar rămânem pe pinul 9, ca în tot modulul.

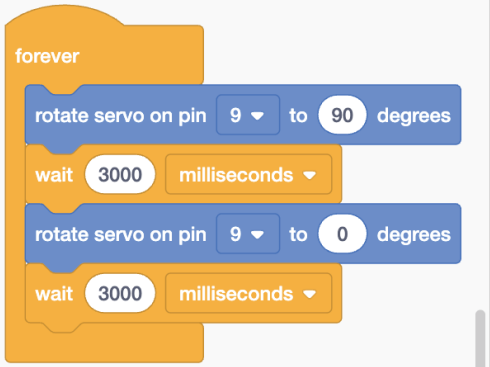


☐ **Cele trei fire sunt la locul lor**

TRASEUL B · ARDUINO UNO
Semnalul → pinul 9 · unghiul se cere direct în grade, din blocuri

2 Blocuri și vitrina Servo.h

Apasă „Code" și alege modul „Blocks + Text". Aici există un bloc dedicat: rotate servo on pin 9 to 90 degrees : ceri unghiul direct, fără nicio traducere. Ciclul: forever → rotate servo on pin 9 to 90 degrees, wait 3000 ms → rotate servo on pin 9 to 0 degrees, wait 3000 ms. În dreapta, Tinkercad adaugă singur biblioteca Servo și scrie servo_9.write(90) : „scrie unghiul", pe scurt.




```cpp
// C++ code
//
#include <Servo.h>

Servo servo_9;

void setup()
{
  servo_9.attach(9, 500, 2500);
}

void loop()
{
  servo_9.write(90);
  delay(3000); // Wait for 3000 millisecond(s)
  servo_9.write(0);
  delay(3000); // Wait for 3000 millisecond(s)
}
```

☐ **Bariera urcă și coboară singură**

C++ GENERAT, CU COMENTARII ÎN LIMBA ROMÂNĂ **C++ · ARDUINO**

TRASEUL B · ARDUINO UNO

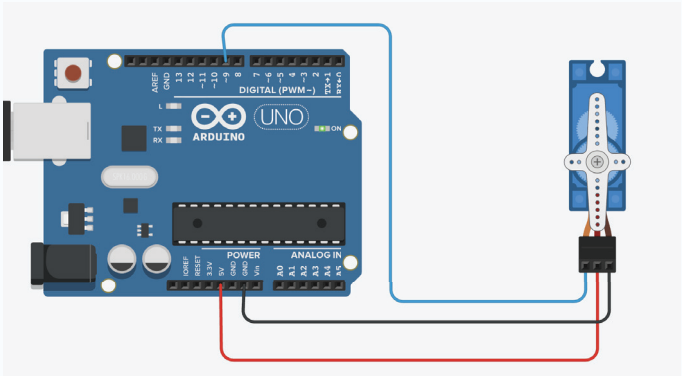
Semnalul → pinul 9 · unghiul se cere direct în grade, din blocuri

C++ GENERAT, CU COMENTARII ÎN LIMBA ROMÂNĂ

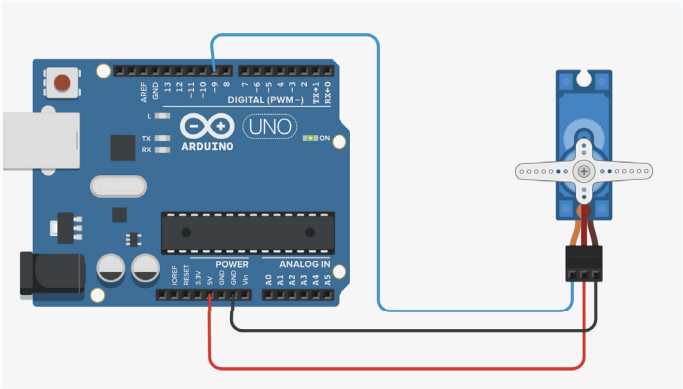
C++ · ARDUINO

```
// C++ code
// Lecția 2.6: bariera de parcare (varianta Arduino)
#include <Servo.h>   // biblioteca de servomotoare: aduce comenzile gata scrise
Servo servo_9;   // dăm un nume servomotorului de pe pinul 9
void setup() {   // se execută o singură dată, la pornire
  servo_9.attach(9, 500, 2500);   // leagă numele de pinul 9 și de plaja de impulsuri
}
void loop() {   // se repetă la nesfârșit
  // write() cere direct unghiul în grade; biblioteca face singură traducerea
  // în lățimi de impuls, adică exact calculul pe care îl faci de mână la Micro:bit
  servo_9.write(90);   // 90° → bariera sus
  delay(3000);   // ține bariera ridicată 3 secunde
  servo_9.write(0);   // 0° → bariera jos
  delay(3000);   // pauză, apoi de la capăt
}
```

Așa arată bariera în cele două poziții ale ciclului:



a doua imagine: Bariera la 90°: ridicată



prima imagine: Bariera la 0°: coborâtă

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT

Traduci gradele în valori de impuls: mai multă muncă, dar vezi exact ce mesaj primește servomotorul.

ARDUINO UNO

Ceri unghiul direct în grade, dintr-un singur bloc: cel mai comod mod de a comanda un servomotor.

Concluzia: pentru servomotoare, Arduino este drumul scurt, iar Micro:bit este drumul care te învață mai mult. Amândouă ajung la aceeași barieră ridicată.

SECȚIUNEA 03 · DEPANARE

⚠ Servomotorul tremură sau nu ajunge la unghi?

1) Firul roșu este la alimentare (3V / 5V), nu într-un pin de comandă: semnalul comandă, alimentarea furnizează energia. 2) La Micro:bit: ai setat perioada de 20 ms înainte de prima scriere? 3) Valorile sunt în plaja 26–128? În afara ei, servomotorul primește mesaje fără sens. 4) La Arduino: unghiul este între 0 și 180? Nu există 270° pe un servomotor obișnuit.

Semnal ≠ alimentare

Firul portocaliu poartă comanda; firul roșu furnizează energia servomotorului.

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă? (8 minute, pe perechi)

Fiecare pereche își prezintă bariera în 30 de secunde: ce face, ce unghi folosește, un lucru care nu a funcționat din prima și cum l-au reparat. Urmează runda de precizie: profesorul strigă un unghi, „30 de grade!”, „135!”, iar câștigă perechea care își duce prima cornul exact acolo (la Micro:bit: cine calculează primul valoarea corectă de impuls, prin proporție între 26 și 128). Este și prima repetiție pentru Modulul 5, unde prezentarea proiectului devine materie de studiu.

Varianta Kahoot: recapitularea întregului modul: 10 întrebări din lecțiile 2.1-2.6; linkul și codul QR se află în folderul lecției

Varianta pe foaie: fișa 2.6-J: schița barierei de completat; unghiuri, fire, valorile pentru trei poziții date

Varianta fără ecrane: runda de precizie se joacă și fără ecrane: unghiurile se arată cu brațul, clasa este raportorul

Provocare: trei niveluri, fără pași

Nivel 1 · Trecerea de cale ferată. Adaugă un LED roșu (cu rezistorul lui de 220 Ω, pe alt pin) care clipește cât timp bariera este jos: totul din Modulul 1 revine în joc.

Nivel 2 · Coborârea lină. Barierele reale nu trântesc. Coboară în trepte: 90° → 60° → 30° → 0°, cu pauze scurte între ele.

Nivel 3 · Întrebarea care deschide Modulul 3. Bariera ta funcționează singură, pe bază de timp. Cum ai face-o însă să se ridice doar când apeși un buton (apeși, urcă; apeși din nou, coboară)? Cu exact această întrebare începe lecția următoare: robotul care ascultă.

3 întrebări rapide

1. Prin ce se deosebește servomotorul de motorul DC din Lecția 2.1?
2. Ce codifică lățimea impulsului trimis servomotorului și la ce interval se repetă impulsurile?
3. De ce alimentarea servomotorului vine din 3V / 5V, iar în pin intră doar firul de semnal?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

servomotor: motor care primește un unghi (0–180°) și îl menține, prin control în buclă închisă

corn: brațul de plastic de pe axul servomotorului, pe care prinzi mecanismul

semnal de comandă: firul (portocaliu) pe care vin impulsurile cu unghiul

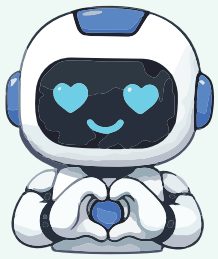
lățimea impulsului : durata „pornit" a impulsului: ~0,5 ms = 0°, ~1,5 ms = 90°, ~2,5 ms = 180°

motor pas cu pas: motor care se rotește în pași ficși (tipic 1,8°); mișcă imprimantele 3D

mecanism: ansamblu de piese în mișcare care îndeplinesc împreună o funcție utilă; bariera ta

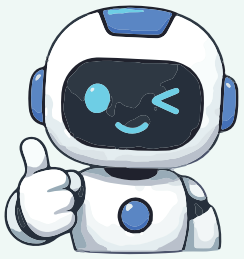
Ce trebuie să știi după această lecție?

- ✓ Servomotor = motor + reductor + control de poziție; primește unghi, îl ține
- ✓ Mesajul este un impuls la fiecare 20 ms; lățimea lui (0,5–2,5 ms) codifică unghiul
- ✓ Semnalul intră în pin; forța vine din alimentare, niciodată invers
- ✓ Bariera ta este primul mecanism complet: program → semnal → mișcare utilă



Ai terminat Modulul 2!

Acum șase ore, „motor” însemna o cutie misterioasă. Astăzi știi ce se rotește în interior, protejezi tranzistorul cu o diodă, inversezi sensul dintr-o pereche de instrucțiuni, reglezi viteza în procente și comanzi un unghi la grad. Un singur lucru îi lipsește robotului tău: nu știe ce se întâmplă în jurul lui. În Modulul 3 primește simțuri (butoane, lumină, temperatură, distanță) și învață cea mai importantă construcție din programare: „dacă... atunci...”. Ne vedem acolo!



MODULUL 2 · ANEXA ELEVULUI

Răspunsuri la „3 întrebări rapide”

Lecția 2.1

1. Actuatorul transformă energia electrică într-un efect fizic (mișcare, lumină, sunet). Exemple: difuzorul, becul, motorul mașinii de spălat, vibrația telefonului.

2. Se rotește în sens invers, cu aceeași viteză: ai inversat polaritatea, deci curentul traversează bobina în sens opus.

3. Turația: numărul de rotații pe minut (rot/min; în simulator, RPM).

Lecția 2.2

1. Turația (n) arată cât de repede se rotește axul; cuplul (M) arată momentul forței cu care răsucește; se măsoară în rot/min, respectiv N·m.

2. i = 45 ÷ 15 = 3; turația scade de 3 ori (iar cuplul crește de circa 3 ori).

3. Motorul simplu este rapid, dar prea slab pentru a urni roțile; reductorul schimbă surplusul de viteză pe forța necesară.

Lecția 2.3

1. Motorul cere sute de miliamperi; pinul suportă câțiva miliamperi (micro:bit) sau aproximativ 20 mA (Arduino). Conectat direct la pin, motorul l-ar distruge.

2. Curentul mic de la bază deschide traseul principal colector → emitor; tranzistorul este un comutator comandat electric.

3. Absoarbe impulsul de tensiune produs prin autoinducție de bobina motorului la oprire; dunga (catodul) privește spre alimentare.

Anexa continuă pe pagina următoare cu Lecțiile 2.4-2.6.

MODULUL 2 · ANEXA ELEVULUI

Răspunsuri la „3 întrebări rapide”

Lecția 2.4

1. Înainte: S1 + S4 (diagonala). Înapoi: S2 + S3 (cealaltă diagonală).
2. Două comutatoare de pe aceeași parte creează un traseu direct alimentare → GND, ocolind motorul: scurtcircuit.
3. Enable este comutatorul general al punții: fără 1 pe el, ieșirile rămân inactive, indiferent de intrări.

Lecția 2.5

1. Procentul din perioadă în care pinul este pornit: timp pornit ÷ perioadă × 100 %.
2. 25 % din 1023 ≈ 256 (micro:bit); 25 % din 255 ≈ 64 (Arduino).
3. Motorul are inerție și frecare: sub un anumit cuplu nu se urnește; LED-ul nu are piese în mișcare, deci luminează și la impulsuri slabe.

Lecția 2.6

1. Motorul DC se rotește liber cât are curent; servomotorul primește un unghi (0–180°), se duce acolo și ține poziția.
2. Lățimea impulsului codifică unghiul (~0,5 ms = 0°, ~1,5 ms = 90°, ~2,5 ms = 180°); impulsurile se repetă la fiecare 20 ms.
3. Pinul este poartă de comandă, nu de forță: semnalul este de mică putere, iar motorul din servomotor cere un curent considerabil, pe care îl ia din alimentare.

Ai terminat Modulul 2!

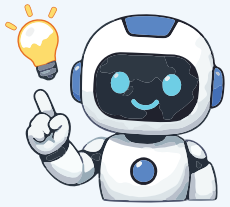
Răspunsurile de mai sus încheie Anexa elevului pentru Modulul 2.

Modulul 3

SENZORI

și achiziția datelor

Modulul 3. Senzori și achiziția datelor



Robotul tău știe deja să lumineze și să miște. Dar încă nu află nimic despre lumea din jur. Modulul 3 îi dă simțuri (butoane, lumină, temperatură, distanță, mișcare) și îl învață cel mai important cuvânt din programare: „dacă”. Un telefon are în jur de zece senzori; un automobil modern, câteva sute. Fiecare face exact ce înveți tu în următoarele șase lecții: transformă o mărime fizică într-un număr, iar numărul într-o decizie.

Lanțul pe care îl construim în acest modul

fenomen fizic → semnal electric → număr → decizie → acțiune

Lumina cade pe senzor (fenomen), senzorul o transformă în tensiune (semnal), convertorul plăcii o transformă în număr, programul compară numărul cu un prag (decizie), iar LED-ul sau motorul primesc comanda (acțiune). Fiecare lecție adaugă câte o verigă; în 3.6 lanțul este complet, cu doi senzori care lucrează împreună.

LEGENDA CULORILOR · ACELEAȘI ÎN TOT MANUALUL

Teorie · Practică Tinkercad · Atenție / Depanare · Provocare / Joc

LECȚIA 3.1 — **Robotul ascultă: intrări digitale și prima decizie**

Pinii ca intrări · butonul · instrucțiunea „dacă… atunci… altfel”

LECȚIA 3.2 — **Semnale care variază: potențiometrul și convertorul analog-digital**

Analogic vs. digital · scara 0–1023 · afișarea valorilor

LECȚIA 3.3 — **Lumina: fotorezistorul și noțiunea de prag**

Împărțitorul de tensiune · pragul · lampa care se aprinde singură

LECȚIA 3.4 — **Temperatura: senzorul TMP36 și citirile cu zgomot**

Din număr în grade · variabilele · media care netezește zgomotul

LECȚIA 3.5 — **Distanța și mișcarea: senzorul ultrasonic și senzorul PIR**

Principiul ecoului · senzorul de parcare · alarma de mișcare

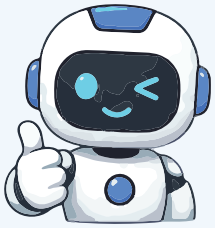
LECȚIA 3.6 — **Senzorii roboților adevărați și fuziunea senzorilor**

Atlasul senzorilor: camere, LiDAR, accelerometru, GPS · doi senzori, o decizie

LECȚIA 3.1 · 50 DE MINUTE

Robotul ascultă: intrări digitale și prima decizie

Până astăzi, placa doar transmitea comenzi prin pini. De astăzi, și ascultă. Înveți să citești un buton și scrii prima instrucțiune care ia o decizie: „dacă... atunci... altfel”.



50 MIN · DIFICULTATE ●●○ · PRIMA DECIZIE DIN COD · AMBELE TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

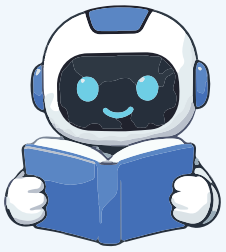
- Să explici diferența dintre un pin folosit ca ieșire și unul folosit ca intrare
- Să citești starea unui buton din program
- Să scrii o instrucțiune „dacă... atunci... altfel”, în blocuri și în Python

SECȚIUNEA 01 · TEORIE

Pinii au două sensuri

DEJA CUNOȘTI (DIN LECȚIILE 1.6 ȘI 2.6)

LCu blocul de scriere digitală (HIGH / LOW) porneai și opreai un pin: pinul era o **ieșire**. Iar Modulul 2 s-a încheiat cu o întrebare: cum faci bariera să asculte de un buton? Astăzi primești răspunsul.



Orice pin de comandă poate lucra și invers: în loc să scoată curent la comanda ta, poate **citi** ce tensiune este prezentă la capătul lui. Un pin folosit astfel se numește **intrare**. Citirea cea mai simplă este cea **digitală**: pinul răspunde cu una din două valori, 1 (tensiune prezentă) sau 0 (absența tensiunii). Este ideală pentru componentele cu doar două stări, cum este butonul: apăsat sau liber.

Instrucțiunea care decide: „dacă… atunci… altfel”

Un număr citit nu folosește la nimic dacă programul nu poate reacționa la el. Aici intră în scenă cea mai importantă instrucțiune din toată programarea, prezentă în orice limbaj din lume: **instrucțiunea condițională**. Ea are trei părți: o **condiție** , adică o expresie logică ce poate fi doar adevărată sau falsă (de exemplu: „butonul este apăsat”); ramura **atunci** , cu instrucțiunile executate când condiția este adevărată; și ramura **altfel** , cu instrucțiunile executate când condiția este falsă. Programul alege întotdeauna exact una dintre ramuri, niciodată pe amândouă.

Anatomia instrucțiunii, în cele două haine

În blocuri: piesa „if… then… else”: o carcasă cu două buzunare; condiția se așază în capul piesei, iar instrucțiunile în buzunare.

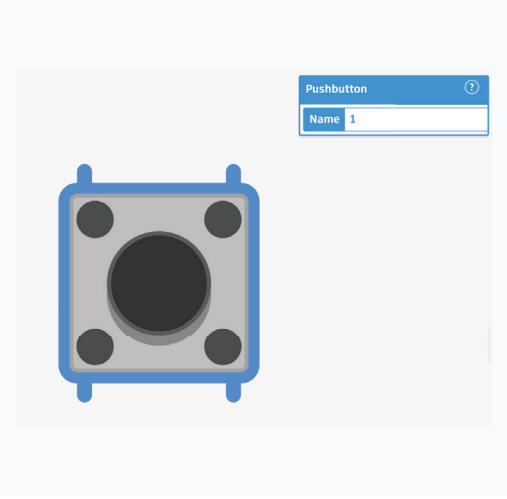
În Python: rândul începe cu if, urmează condiția și cele două puncte (:); ramura „atunci” stă dedesubt, împinsă cu 4 spații, adică indentarea pe care o cunoști deja. Ramura „altfel” începe cu else: și are propriile rânduri indentate.

Regula lecturii cu voce tare: „DACĂ este apăsat butonul, ATUNCI aprinde LED-ul, ALTFEL stinge-l.” Dacă propoziția sună logic în limba română, este scrisă corect și în cod.

SECȚIUNEA 02 · PIESA DE AZI

Piesă nouă: Butonul (pushbutton)

PRIMA APARIȚIE



CE ESTE

O componentă cu o pastilă care se apasă și patru terminale (conectate două câte două). Cât timp pastila este apăsată, cele două perechi de terminale sunt unite; când o eliberezi, legătura se întrerupe.

CE FACE

Este un întrerupător de moment: închide circuitul doar cât îl ții apăsat. Conectat între alimentare și un pin-intrare, îi transmite pinului 1 la apăsare și 0 în rest, cu o condiție importantă, explicată mai jos: rezistorul de referință.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ

Tastatura, soneria, butoanele lifturilor și ale telecomenzilor: cea mai răspândită componentă de intrare din electronică.

DE CE NE TREBUIE AZI

Este cel mai simplu senzor posibil: două stări, zero reglaje; ideal pentru prima citire și prima decizie.

Rezistorul de referință (pull-down)

Un pin-intrare lăsat „în aer”, neconectat la nimic, nu citește 0: citește la întâmplare, ca un radio între posturi. De aceea, la butonul extern, între pin și GND se montează un rezistor de 10 kΩ, numit **rezistor de referință** (în limba engleză: pull-down): el menține pinul ferm pe 0 cât timp butonul este liber. Apeși butonul → tensiunea de alimentare ajunge la pin → citirea devine 1. Fără referință, citirile ar fi haotice; aceasta este prima regulă a intrărilor.

SECȚIUNEA 03 · PRACTICĂ

Apeși: se aprinde. Eliberezi: se stinge.

Construcția reia LED-ul cu rezistorul lui de protecție din Lecția 1.6 și adaugă intrarea. Logica, identică pe ambele trasee: **dacă butonul e apăsat, LED-ul e aprins; altfel, stins.**

TRASEUL A · MICRO:BIT

Butonul A este deja pe placă, fără cablare pentru intrare · LED-ul pe P0

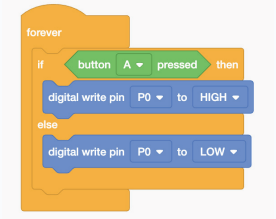
CIRCUIT DE START

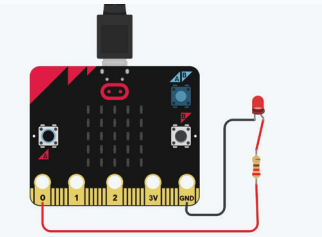
▶ Deschide circuitul de start 3.1-A (Micro:bit)

LED-ul cu rezistorul de 220 Ω, gata conectate la P0 și GND (circuitul din 1.6). Apasă „Copy and Tinker”.

1 · Scrie decizia: întâi blocuri, apoi Python

Placa ta are butoanele A și B încorporate: le citești **direct din cod**, fără niciun fir. În blocuri: forever → if button A is pressed → digital write pin P0 to HIGH, else → digital write pin P0 to LOW. Rulează: ține apăsat butonul A din simulator și privește LED-ul. Treci apoi pe Python; programul este mai jos, rând cu rând.





☐ LED-ul ascultă de butonul A

PRIMA DECIZIE · VARIANTA ÎN TEXT — PYTHON · MICRO:BIT

```
# Lecția 3.1: LED comandat de buton
# (varianta Tinkercad / MakeCode)

def on_forever():   # blocul pe care placa îl repetă la nesfârșit
    if input.button_is_pressed(Button.A):   # condiția: e apăsat butonul A?
        pins.digital_write_pin(DigitalPin.P0, 1)   # atunci: LED aprins
    else:   # altfel:
        pins.digital_write_pin(DigitalPin.P0, 0)   # LED stins

basic.forever(on_forever)   # forever din blocuri = acest rând în Python
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Buton extern pe pinul 2, cu rezistor de referință de 10 kΩ · LED-ul pe pinul 13

CIRCUIT DE START

▶ Deschide circuitul de start 3.1-B (Arduino)

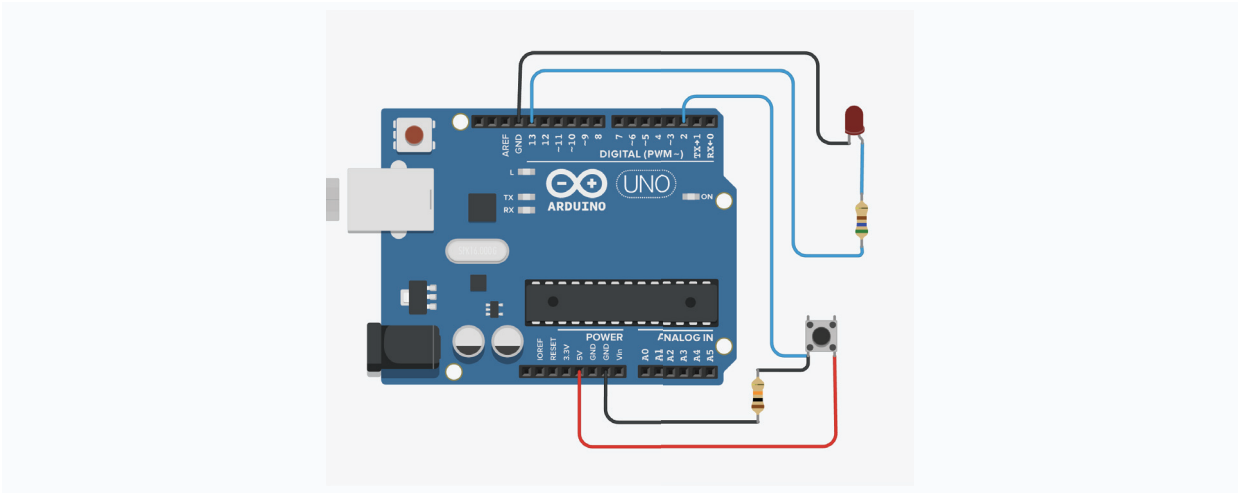
LED-ul cu rezistorul de 560 Ω pe pinul 13 (circuitul din 1.6), plus butonul și rezistorul de 10 kΩ, neconectate. Apasă „Copy and Tinker”.

1 · Cablează intrarea

Un terminal al butonului → pinul **5V**. Terminalul-pereche opus → pinul **2** și, tot de acolo, rezistorul de **10 kΩ → GND**. Ai construit exact schema din teorie: alimentare → buton → pin, cu referința care menține pinul pe 0.

1 · Cablează intrarea

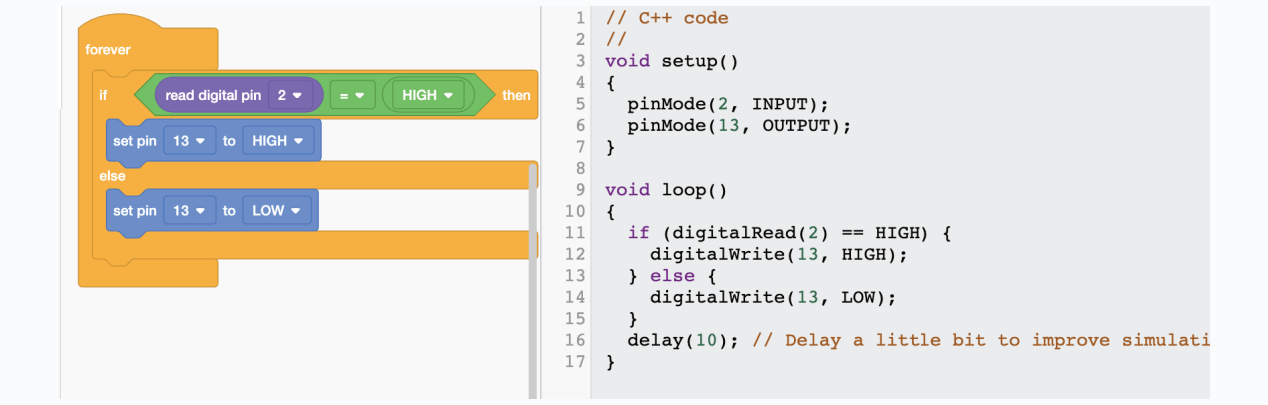
Un terminal al butonului → pinul **5V**. Terminalul-pereche opus → pinul **2** și, tot de acolo, rezistorul de **10 kΩ → GND**. Ai construit exact schema din teorie: alimentare → buton → pin, cu referința care menține pinul pe 0.



☐ Intrarea este cablată

2 · Scrie decizia în blocuri și privește vitrina

În blocuri: forever → if read digital pin 2 is HIGH → set pin 13 to HIGH, else → set pin 13 to LOW. Rulează și apasă butonul din simulator. În vitrina C++ apare perechea nouă: digitalRead (citirea) și if / else, cu aceleași trei părți: condiție, atunci, altfel.




```
// C++ code
//
void setup()
{
  pinMode(2, INPUT);
  pinMode(13, OUTPUT);
}

void loop()
{
  if (digitalRead(2) == HIGH) {
    digitalWrite(13, HIGH);
  } else {
    digitalWrite(13, LOW);
  }
  delay(10); // Delay a little bit to improve simulati
}
```

☐ LED-ul ascultă de buton

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ C++ · ARDUINO

```
// C++ code
// Lecția 3.1: LED comandat de buton (varianta Arduino)

void setup() {   // se execută o singură dată, la pornire
  pinMode(2, INPUT);   // pinul 2 devine intrare: aici citim butonul
  pinMode(13, OUTPUT);   // pinul 13 devine ieșire: aici comandăm LED-ul
}

void loop() {   // se repetă la nesfârșit
  // digitalRead întoarce HIGH (1) sau LOW (0), exact ce transmite butonul
  if (digitalRead(2) == HIGH) {   // condiția: e apăsat butonul?
    digitalWrite(13, HIGH);   // atunci: LED aprins
  } else {
    digitalWrite(13, LOW);   // altfel: stins
  }
  delay(10);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT
Două butoane deja pe placă: prima decizie merge fără niciun fir. Cel mai rapid start posibil.

ARDUINO UNO
Butonul se cablează manual, cu rezistor de referință: mai multă muncă, dar exact așa se conectează orice buton din lumea reală.

Concluzia: începi mai ușor pe micro:bit, dar cablarea de la Arduino este lecția care rămâne: intrare = semnal + referință. Pe oricare traseu, instrucțiunea „dacă” este identică.

SECȚIUNEA 04 · DEPANARE

⚠ Decizia nu funcționează?

1) La Arduino: LED-ul se aprinde singur, la întâmplare? Lipsește rezistorul de referință spre GND: pinul „plutește”. 2) Butonul nu face nimic? Verifică terminalele-perechi: cele două fire trebuie conectate pe terminale opuse, nu pe aceeași pereche. 3) În Python: else: are propriile două puncte și propriile rânduri indentate? 4) Condiția citește exact pinul la care este conectat butonul?

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă (6 minute, toată clasa)

Profesorul dă comenzi cu condiție, iar elevii le execută doar dacă propoziția este adevărată pentru ei: „dacă porți ceva albastru, ridică mâna dreaptă” · „dacă ziua ta este într-o lună cu 31 de zile, ridică-te” · „dacă ai frați, atinge banca; altfel, încrucișează brațele” (prima comandă cu altfel: toată clasa execută ceva!). Cine execută greșit iese o rundă. Ultimele runde: condiții din lecție, „dacă valoarea citită este 1…”, iar clasa observă că jocul este chiar instrucțiunea if, jucată pe oameni.

Varianta Kahoot: condiții de evaluat: „adevărat sau fals?” pe situații date; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 3.1-J: completează cele trei părți (condiție / atunci / altfel) pentru 4 situații din viața reală
Varianta fără ecrane: chiar jocul principal; necesită doar voce și atenție

Provocare: răspunsul la întrebarea din Modulul 2

Bariera cu buton. Adaugă servomotorul din Lecția 2.6 pe planșă (semnalul pe P1, respectiv pinul 9) și rescrie decizia: dacă butonul este apăsat, bariera stă la 90°; altfel, la 0°. Cât timp ții apăsat, bariera este sus: prima ta comandă de mecanism printr-o intrare.

3 întrebări rapide

1. Prin ce se deosebește un pin-intrare de un pin-ieșire?

2. Din ce trei părți este alcătuită instrucțiunea condițională și câte ramuri se execută?

3. La ce folosește rezistorul de referință de 10 kΩ al butonului?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

intrare / ieșire: pin care citește semnale / pin care scoate semnale

citire digitală: citire cu două valori posibile: 1 (tensiune prezentă) sau 0 (absența tensiunii)

instrucțiune condițională: „dacă… atunci… altfel”; programul alege o singură ramură

condiție: expresie logică ce poate fi doar adevărată sau falsă

buton (pushbutton): întrerupător de moment: unește contactele doar cât este apăsat

rezistor de referință (pull-down): rezistorul de 10 kΩ care menține pinul-intrare ferm pe 0

read_digital / digitalRead: comenzile de citire digitală pe cele două plăci

Ce trebuie să știi după această lecție

- ✓ Pinii au două sensuri: scriu (ieșire) sau citesc (intrare)
- ✓ „Dacă… atunci… altfel”: o condiție, două ramuri, se execută exact una
- ✓ Butonul extern se conectează întotdeauna cu rezistor de referință spre GND
- ✓ Bariera din Modulul 2 ascultă acum de un buton: primul tău lanț simte → decide → acționează

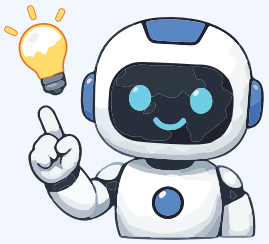
Primul lanț complet din Modulul 3

LECȚIA 3.2 · 50 DE MINUTE

Semnale care variază: potențiometrul și convertorul analog-digital

Butonul cunoaște doar „da” și „nu”. Lumea este însă plină de mărimi care variază continuu: lumina, temperatura, volumul. În această lecție înveți cum le transformă placa în numere.

50 MIN · DIFICULTATE ●●○ · PRIMA CITIRE ANALOGICĂ · AMBELE TRASEE



CE VEI ȘTI LA FINALUL LECȚIEI

- Să explici diferența dintre un semnal digital și unul analogic
- Să explici ce face convertorul analog-digital și pe ce scară răspunde
- Să citești un potențiometru și să afișezi valoarea, pe traseul tău

SECȚIUNEA 01 · TEORIE

Analogic și digital

DEJA CUNOȘTI (DIN LECȚIILE 3.1 ȘI 2.5)

Citirea digitală răspunde cu 1 sau 0. Iar de la PWM cunoști deja ideea de scară de valori: acolo scriai pe ea; astăzi vei citi de pe una.

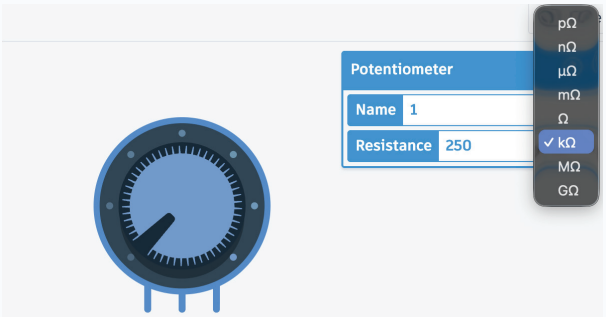
Un **semnal digital** are trepte fixe: 0 sau 1, apăsat sau liber. Un **semnal analogic** poate lua orice valoare dintr-un interval, fără trepte, precum temperatura de afară sau lumina dimineții, care nu sar de la „nimic” la „totul”, ci cresc continuu. Senzorii analogici traduc asemenea mărimi într-o tensiune care variază la fel de continuu.

Procesorul lucrează însă doar cu numere. Traducerea este realizată de un circuit din interiorul plăcii: **convertorul analog-digital** (prescurtat ADC, din limba engleză: Analog-to-Digital Converter). El măsoară tensiunea de pe pin și o transformă într-un număr întreg, printr-o simplă proporție: 0 volți → 0; tensiunea maximă a plăcii (3,3 V la micro:bit, 5 V la Arduino) → 1023. Jumătate din tensiune → aproximativ 512. De unde vine exact 1023? Convertorul ambelor plăci lucrează pe 10 biți, adică distinge 2^{10} = 1024 de trepte, numerotate de la 0 la 1023. Reține detaliul onest: la citire, ambele plăci folosesc aceeași scară, 0–1023 ; scara diferită din Modulul 2 (0–255) era doar la scriere, la Arduino.

SECȚIUNEA 02 · PIESA DE AZI

Piesă nouă: Potențiometrul

PRIMA APARIȚIE



CE ESTE

Un rezistor reglabil cu trei terminale: două capete și, între ele, un cursor care alunecă pe pista rezistivă atunci când rotești axul.

CE FACE

Conectat cu un capăt la alimentare și cu celălalt la GND, funcționează ca un împărțitor de tensiune: cursorul culege o tensiune între 0 și maxim, proporțională cu rotația. Este cel mai curat semnal analogic de laborator; îl controlezi cu mâna.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ

Butonul de volum al boxelor, reglajele consolelor de mixaj, pedala de accelerație a multor automobile: toate sunt potențiometre.

DE CE NE TREBUIE AZI

Pentru a vedea convertorul la lucru: rotești lin cu mâna și privești numărul urcând de la 0 la 1023.

SECȚIUNEA 03 · PRACTICĂ

Rotești și citești: 0 … 1023

Cablajul, comun: capătul 1 al potențiometrului → alimentare (3V / 5V); capătul 2 → GND; cursorul (piciorul din mijloc) → pinul analogic. Diferă doar locul unde vezi numărul; exact aici cele două plăci își arată personalitatea.

TRASEUL A · MICRO:BIT

Cursorul → P0 · numărul defilează pe afișajul cu 25 de LED-uri

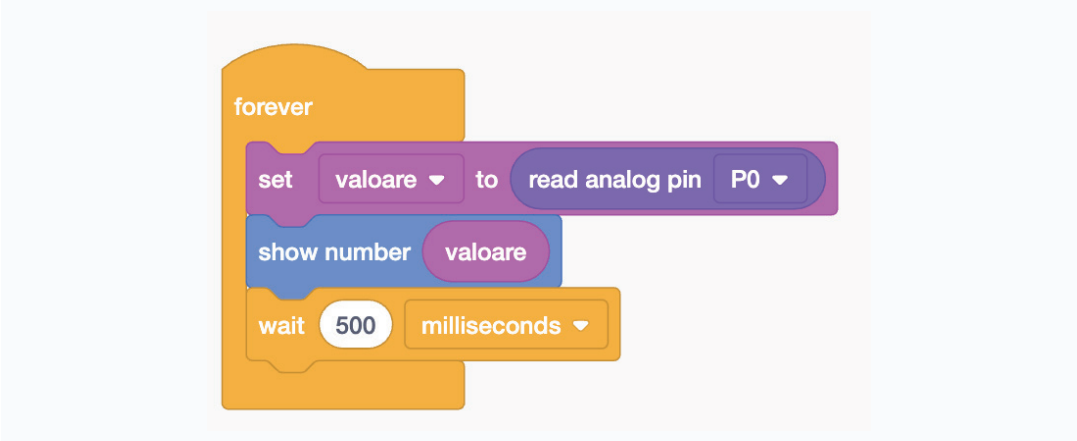
CIRCUIT DE START

▶ Deschide circuitul de start 3.2-A (Micro:bit)

Placa și potențiometrul se află pe planșa de lucru. Apasă „Copy and Tinker”, conectează cele trei fire și treci la cod.

1 · Construiește programul în blocuri

Apasă „Code”. Din sertare: forever → set valoare to (read analog pin P0) → show number (valoare) → wait 500 milliseconds. Blocul set ... to vine din sertarul Variables, după ce apeși „Make a Variable” și îi dai numele valoare. Comută apoi pe Python: același program, rând cu rând.



☐ Blocurile sunt îmbinate și programul este gata de rulat

POTENȚIOMETRUL · VARIANTA ÎN TEXT **PYTHON · MICRO:BIT**

```
# Lecția 3.2: citim potențiometrul
# (varianta Tinkercad / MakeCode)

def on_forever():
    valoare = pins.analog_read_pin(AnalogPin.P0)   # numărul de la convertor: 0–1023
    basic.show_number(valoare)   # îl afișăm pe matricea de 25 de LED-uri
    basic.pause(500)

basic.forever(on_forever)
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Cursorul → A0 · numerele curg în Monitorul serial

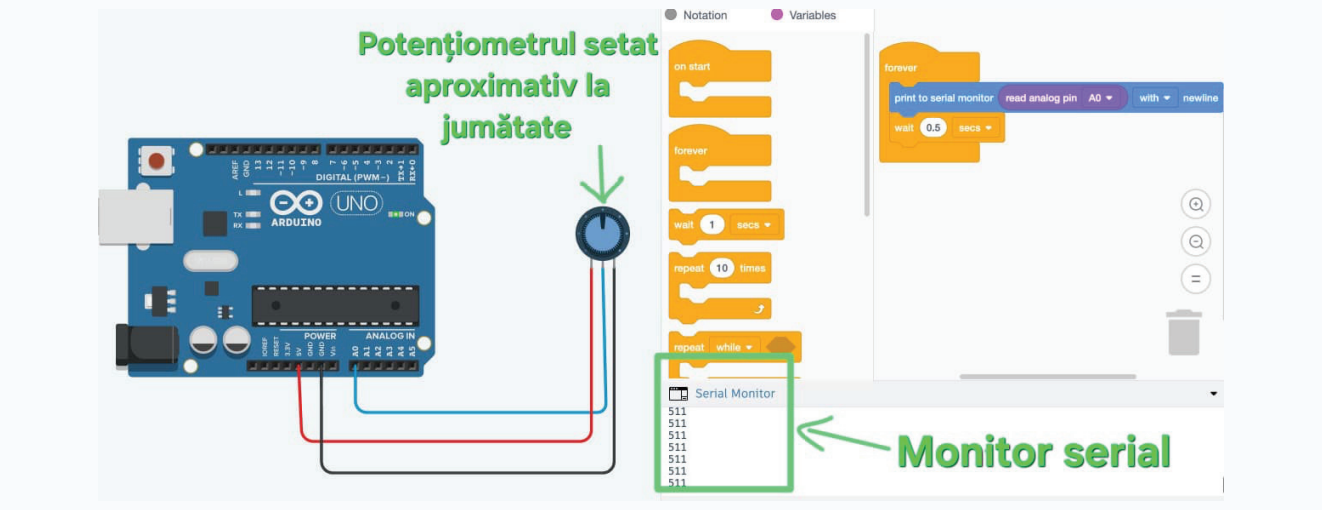
CIRCUIT DE START

▶ Deschide circuitul de start 3.2-B (Arduino)

Placa și potențiometrul. Apasă „Copy and Tinker”, conectează cele trei fire (cursorul la A0) și treci la cod.

1 · Citește în blocuri și deschide Monitorul serial

Pinii A0–A5 sunt intrările analogice ale plăcii Arduino; de aceea poartă litera A. În blocuri: forever → print to serial monitor (read analog pin A0), wait 0.5 secs. Apasă apoi butonul Serial Monitor din partea din stânga jos a zonei de cod: se deschide „jurnalul de date” al plăcii, în care valorile curg una sub alta în timp ce rotești. În vitrina C++: analogRead(A0) și Serial.println.



☐ Valorile curg în Monitorul serial

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ — C++ · ARDUINO

```
// C++ code
// Lecția 3.2: citim potențiometrul (varianta Arduino)

void setup() {
  Serial.begin(9600);   // deschide Monitorul serial
  pinMode(A0, INPUT);
}

void loop() {
  // analogRead citește tensiunea de pe A0 și o întoarce ca număr între 0 și 1023
  Serial.println(analogRead(A0));
  delay(500);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT
Numărul apare chiar pe placă, fără instrumente suplimentare; cifrele defilează însă încet, valoare cu valoare.

ARDUINO UNO
Monitorul serial este un jurnal întreg: sute de valori pe minut, ușor de urmărit și de comparat; instrumentul serios de măsură.

Concluzia: pentru verificări rapide, afișajul micro:bit este suficient; pentru măsurători adevărate, Monitorul serial de la Arduino este fără rival. Îl vei folosi des de acum înainte.

SECȚIUNEA 04 · DEPANARE

⚠ Numărul stă pe loc sau sare haotic?

1) Cursorul (terminalul din mijloc!) este conectat exact la pinul analogic? Capetele la alimentare și GND? 2) La Arduino: citirea este pe A0–A5, nu pe un pin digital? 3) Numărul stă blocat pe 0 sau 1023 → un capăt este neconectat. 4) Ai rotit efectiv potențiometrul în simulare? (Glisorul apare doar cu simularea pornită.)

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă (6 minute, pe echipe)

Runda 1: profesorul rotește potențiometrul pe proiector și îngheață o poziție; echipele scriu estimarea numărului (0–1023) pe foaie, apoi valoarea reală se citește pe ecran. Cea mai apropiată estimare ia punctul. Runda 2, invers: profesorul strigă o țintă, „750!”, iar echipele spun unde trebuie adus axul (aproximativ trei sferturi). Runda finală: țintele devin procente, „25 %!”, iar clasa face conversia mental: 25 % din 1023 ≈ 256. Proporțiile din Modulul 2, întoarse pe partea de citire.

Varianta Kahoot: poziții de potențiometru de estimat, plus conversii procente ↔ valori; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 3.2-J: scara 0–1023 desenată; marchează pozițiile cerute și completează conversiile
Varianta fără ecrane: chiar jocul principal; este suficient un singur calculator la proiector

Provocare: al doilea senzor analogic

Senzorul de forță. Caută în dulapul cu piese „Force sensor” și conectează-l în locul potențiometrului: are doar două terminale, unul la alimentare, celălalt la pin și, prin rezistorul de 10 kΩ, la GND. Este același principiu pe care îl vei învăța în lecția următoare: două rezistențe în serie, cu citirea de la mijloc. Apasă-l în simulare. Cântarele electronice și ecranele sensibile la apăsare încep exact așa.

3 întrebări rapide

1. Prin ce se deosebește un semnal analogic de unul digital?
2. Ce face convertorul analog-digital și ce număr dă pentru jumătate din tensiunea maximă?
3. Pe ce scară citesc cele două plăci și unde era, de fapt, diferența 0–255 din Modulul 2?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

semnal analogic: semnal care variază continuu, putând lua orice valoare dintr-un interval

convertor analog-digital (ADC): circuitul din placă ce transformă tensiunea în număr (0–1023, pe 10 biți)

potențiometru: rezistor reglabil cu cursor; împarte tensiunea proporțional cu rotația

cursor: terminalul din mijloc al potențiometrului; de aici se culege semnalul

analog read pin / analogRead: comenzile de citire analogică pe cele două plăci

Monitor serial: „jurnalul de date” al plăcii Arduino: valorile curg una sub alta

variabilă: o zonă de memorie cu nume, în care programul păstrează o valoare

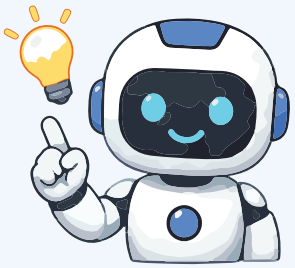
Ce trebuie să știi după această lecție

- ✓ Digital = trepte fixe (0/1); analogic = variație lină, orice valoare din interval
- ✓ Convertorul traduce tensiunea în număr, proporțional: 0 V → 0, maximul plăcii → 1023
- ✓ La citire, ambele plăci folosesc scara 0–1023; 0–255 era doar la scrierea PWM pe Arduino
- ✓ Cursorul potențiometrului intră în pinul analogic; capetele merg la alimentare și GND

LECȚIA 3.3 · 50 DE MINUTE

Lumina: fotorezistorul și noțiunea de prag

Primul senzor „adevărat” din colecție: componenta care simte lumina. Cu ea, cu un prag ales de tine și cu instrucțiunea „dacă”, construiești lampa care se aprinde singură când se întunecă.



50 MIN · DIFICULTATE ●●○ · PRIMUL SENZOR + DECIZIE · AMBELE TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să explici cum simte fotorezistorul lumina și de ce are nevoie de un rezistor-partener
- Să găsești, prin măsurare, pragul care desparte „lumină” de „întuneric”
- Să construiești lampa care se aprinde singură sub prag

SECȚIUNEA 01 · TEORIE

Rezistorul care simte lumina

DEJA CUNOȘTI (DIN LECȚIILE 1.3, 3.1 ȘI 3.2)

Rezistorul limitează curentul (Ω). Instrucțiunea „dacă” alege o ramură. Iar convertorul transformă tensiunea de pe pin într-un număr 0–1023. Astăzi le folosim pe toate trei deodată.

Fotorezistorul (numit și LDR, din limba engleză: Light-Dependent Resistor) este un rezistor cu o proprietate specială: rezistența lui depinde de lumină. În întuneric are rezistență mare (limitează puternic curentul); în lumină puternică, rezistență mică (lasă curentul să treacă aproape liber). Problema: pinul plăcii nu citește rezistențe, ci tensiuni. Pentru a transforma rezistența variabilă într-o tensiune variabilă, îi dăm fotorezistorului un partener fix: montajul se numește împărțitor de tensiune.

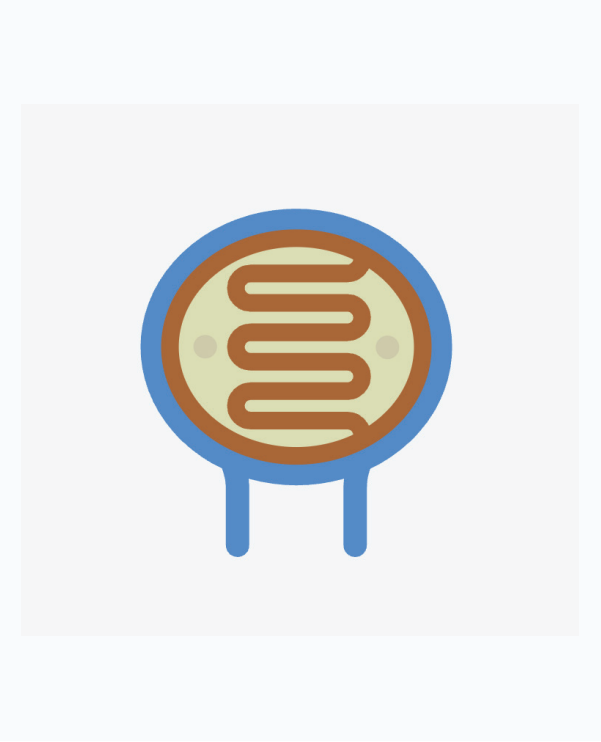
Împărțitorul de tensiune, pe scurt

Fotorezistorul și un rezistor obișnuit de 10 kΩ stau în serie, între alimentare și GND, ca doi consumatori care își împart tensiunea, exact ca în provocarea cu două LED-uri din Lecția 1.4. Pinul citește tensiunea de la mijlocul dintre ei (matematic: Umijloc = Ualimentare × R10kΩ ÷ (Rfoto + R10kΩ)). Când lumina crește, rezistența fotorezistorului scade, punctul de mijloc primește mai multă tensiune, iar numărul citit urcă. În montajul nostru: multă lumină = număr mare, întuneric = număr mic.

SECȚIUNEA 02 · PIESA DE AZI

Piesă nouă: Fotorezistorul

PRIMA APARIȚIE



CE ESTE

O pastilă mică cu două terminale și un traseu șerpuit pe față: stratul fotosensibil care reacționează la lumină. În Tinkercad: „Photoresistor”.

CE FACE

Își schimbă rezistența în funcție de lumina care cade pe el: întuneric → rezistență mare; lumină → rezistență mică. În simulare, execuți clic pe el și îi reglezi „lumina” dintr-un glisor.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ

Iluminatul public care se aprinde singur seara, lămpile de veghe cu senzor, felinarele solare de grădină.

DE CE NE TREBUIE AZI

Este primul senzor care măsoară lumea fără intervenția ta: potențiometrul îl roteai tu; pe acesta îl „rotește” lumina.

Pragul: granița dintre da și nu

Numărul citit variază continuu, dar decizia are doar două trepte: aprind sau nu? Puntea dintre ele este **pragul**: un număr ales de tine, care desparte cele două situații. Sub prag = „întuneric, aprinde”; peste prag = „lumină, stinge”. Pragul nu se ghicește, ci se măsoară: citești valorile în cele două situații reale și alegi un număr între ele. Exact așa își reglează inginerii iluminatul public și luminozitatea automată a telefonului.

SECȚIUNEA 03 · PRACTICĂ

Lampa care se aprinde singură

De aici încolo traseele se despart: același program, în două limbaje. Alege-l pe al tău și scrie-l sau lipește-l din QR-ul de sub cod.

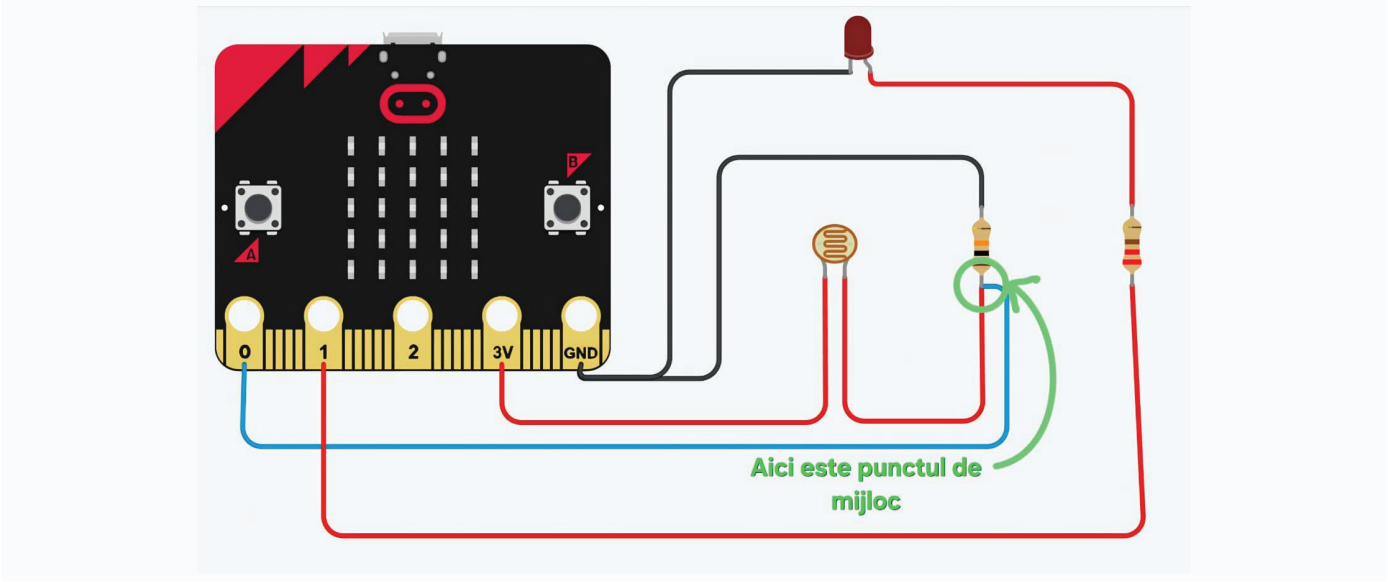


Fig. 1. Punctul de mijloc are două fire: unul coboară spre rezistor, unul pleacă spre pin. Acolo se citește tensiunea împărțită.

CIRCUIT DE START

▶ Deschide circuitul de start 3.3-A (Micro:bit)

Toate componentele pe planșă, împărțitorul neconectat. Apasă „Copy and Tinker”.

CIRCUIT DE START

▶ Deschide circuitul de start 3.3-B (Arduino)

Toate componentele pe planșă, împărțitorul neconectat. Apasă „Copy and Tinker”.

1 · Cablează împărțitorul

Cinci fire, în această ordine:

- alimentare (3V / 5V) → un terminal al fotorezistorului
- celălalt terminal al fotorezistorului → un terminal al rezistorului de 10 kΩ ← acesta este punctul de mijloc
- punctul de mijloc → pinul analogic (P0 / A0)
- celălalt terminal al rezistorului de 10 kΩ → GND
- lampa: P1 / pinul 13 → rezistor de protecție → LED → GND

Al doilea și al treilea fir pleacă din același punct: acolo se citește tensiunea împărțită. Dacă legi pinul direct la alimentare, vei citi mereu 1023, orice ai face cu glisorul de lumină.

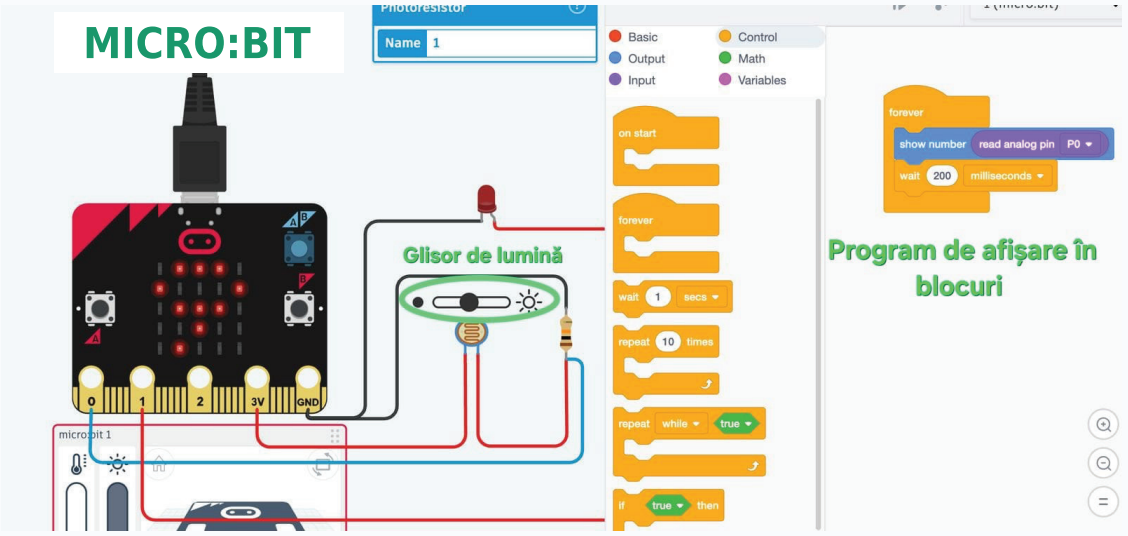
☐ Punctul de mijloc are două fire: unul spre rezistor, unul spre pin

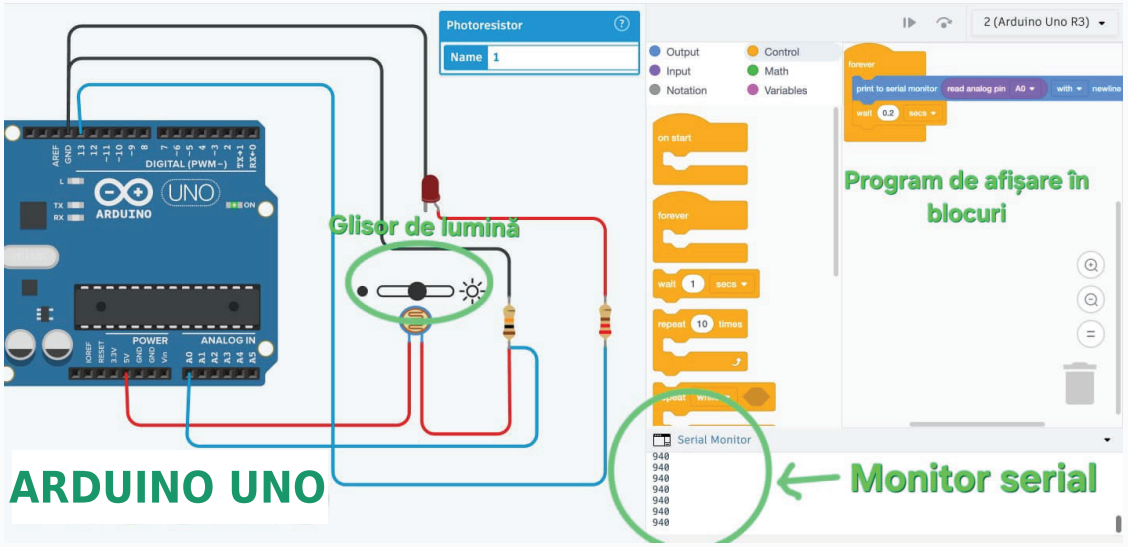
2 · Măsoară-ți pragul

Înainte de decizie ai nevoie de două numere, iar pentru ele îți trebuie un program scurt de afișare, din trei blocuri: forever → show number (read analog pin P0) → wait 200 milliseconds. Pe Arduino, în loc de ecran: forever → print to serial monitor (read analog pin A0) → wait 0.2 secs, apoi deschide Monitorul serial.

Pornește simularea, execută clic pe fotorezistor și trage glisorul de lumină: notează valoarea la lumină plină și valoarea la întuneric. Alege pragul aproximativ la mijlocul dintre ele; de exemplu, dacă ai citit 940 și 54, un prag bun este 400–500.

☐ Am notat cele două valori și mi-am ales pragul



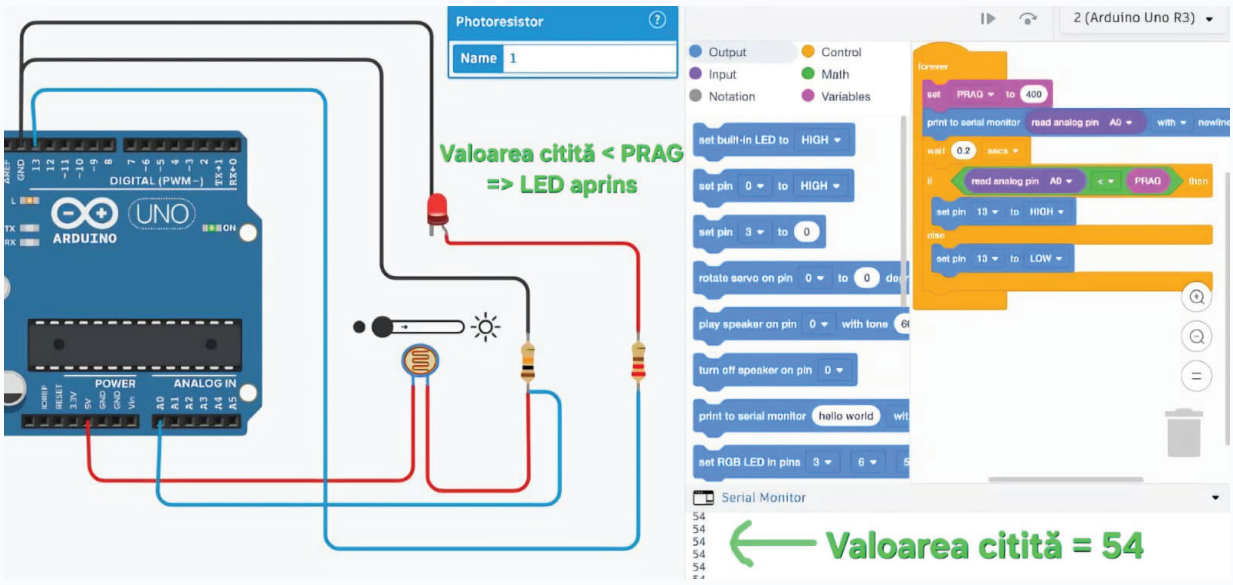


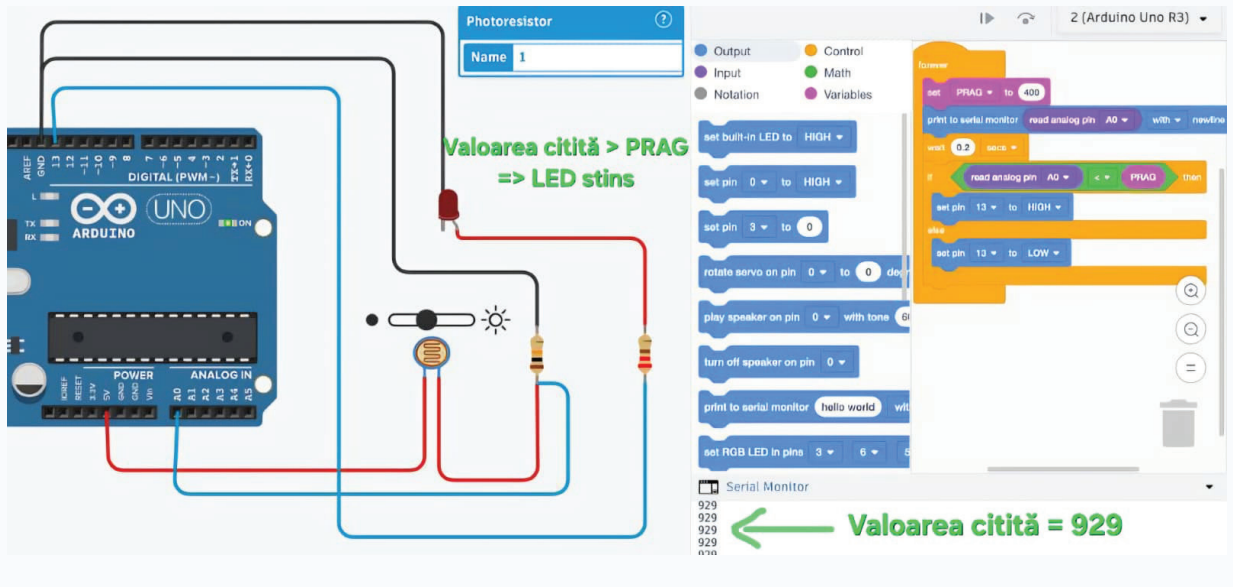
3 · Scrie decizia lămpii

Peste programul de afișare de mai devreme adaugi instrucțiunea „dacă", cu o condiție nouă: compararea cu pragul. La micro:bit, în blocuri: if (read analog pin P0) < PRAG → digital write pin P1 to HIGH, else → digital write pin P1 to LOW. La Arduino, pe blocurile tale: setezi PRAG la 400, afișezi valoarea de pe A0 la fiecare 0.2 secunde, apoi if (read analog pin A0) < PRAG → set pin 13 to HIGH, else → set pin 13 to LOW. Trage glisorul de lumină încet dintr-un capăt în altul și privește lampa, respectiv LED-ul de pe pinul 13, aprinzându-se exact la granița aleasă de tine.

☐ Lampa se aprinde singură la întuneric

Cablajul este identic pe ambele plăci; se schimbă doar numele pinilor și locul în care se scrie codul.





TRASEUL A · MICRO:BIT

Fotorezistorul → P0 · lampa pe P1 · pragul, într-o variabilă

LAMPA DE VEGHE · VARIANTA ÎN TEXT — **PYTHON · MICRO:BIT**

```
# Lecția 3.3: lampa care se aprinde singură
# (varianta Tinkercad / MakeCode)

PRAG = 400   # granița ta, aflată prin măsurare

def on_forever():
    lumina = pins.analog_read_pin(AnalogPin.P0)   # câtă lumină este acum: 0–1023
    if lumina < PRAG:   # este mai întuneric decât granița?
        pins.digital_write_pin(DigitalPin.P1, 1)   # atunci: lampa aprinsă
    else:
        pins.digital_write_pin(DigitalPin.P1, 0)   # altfel: stinsă
    basic.pause(200)

basic.forever(on_forever)
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Fotorezistorul → A0 · lampa pe pinul 13 · pragul, într-o variabilă

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ — C++ · ARDUINO

```
// C++ code
// Lecția 3.3: lampa care se aprinde singură (varianta Arduino)

int PRAG = 400;   // granița ta, aflată prin măsurare
int lumina = 0;

void setup() {
  pinMode(A0, INPUT);
  pinMode(13, OUTPUT);
}

void loop() {
  lumina = analogRead(A0);   // câtă lumină este acum: 0–1023
  if (lumina < PRAG) {   // este mai întuneric decât granița?
    digitalWrite(13, HIGH);   // atunci: lampa aprinsă
  } else {
    digitalWrite(13, LOW);   // altfel: stinsă
  }
  delay(200);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT	ARDUINO UNO
Pragul se schimbă dintr-un singur rând de Python: variabila PRAG, scrisă o dată, folosită peste tot.	Monitorul serial afișează valorile în timp real cât timp cauți pragul; măsurarea este vizibil mai comodă aici.

Concluzia: măsori mai ușor pe Arduino, ajustezi mai elegant pe micro:bit. Iar împărțitorul de tensiune este identic: el ține de electronică, nu de placă.

SECȚIUNEA 04 · DEPANARE

⚠ Valorile nu se mișcă sau lampa nu reacționează?

1) Pinul citește exact punctul de mijloc dintre fotorezistor și rezistorul de 10 kΩ? (Conectat direct la alimentare, citește mereu 1023.) 2) Glisorul de lumină apare doar cu simularea pornită. 3) Lampa „tremură" exact la prag? În simulator nu se întâmplă, valorile stau nemișcate. Pe un montaj real da, iar problema se rezolvă nu cu media, ci cu două praguri în loc de unul: aprinzi sub 380, stingi abia peste 450. Comutarea cu două praguri se numește histerezis și o vei folosi în Modulul 4.

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă (6 minute, pe echipe)

Fiecare echipă primește o situație reală și decide unde pune granița, împreună cu argumentul: iluminatul public al orașului (prag jos, ca să nu funcționeze degeaba ziua? sau mai sus, ca să se aprindă devreme?) · luminozitatea automată a telefonului · lampa de veghe a unui copil · farurile automate ale automobilului în tunel. Echipele își prezintă pragurile, clasa contrazice cu exemple („și pe înserat, când plouă?"). Concluzia care rămâne: pragul este o decizie de proiectare; nu există „numărul corect", ci numărul potrivit situației.

Varianta Kahoot: situații cu praguri de ales, cu 4 variante fiecare; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 3.3-J: patru situații; notează pragul ales și argumentul într-o propoziție
Varianta fără ecrane: chiar jocul principal; dezbatere pe echipe, fără niciun ecran

Provocare: logica întoarsă

Alarma de sertar. Un fotorezistor lipit în sertarul tău stă pe întuneric, până când cineva îl deschide. Rescrie decizia invers: dacă lumina trece PESTE prag, aprinde LED-ul de alarmă. O singură comparație inversată, iar lampa de veghe a devenit sistem de securitate.

3 întrebări rapide

1. Cum își schimbă fotorezistorul rezistența și de ce nu îl putem conecta singur la pin?
2. Ce este pragul și cum îl alegi corect?
3. În montajul nostru, numărul crește sau scade când se face întuneric? De ce?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

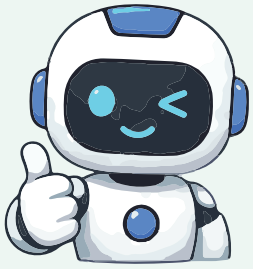
fotorezistor: rezistor (LDR) a cărui rezistență depinde de lumină: întuneric = rezistență mare

împărțitor de tensiune: două rezistențe în serie; pinul citește tensiunea de la mijloc

prag: numărul ales de tine care desparte două situații („lumină” / „întuneric”)

comparație: condiție cu < sau >: „valoarea este mai mică decât pragul?”

Ce trebuie să știi după această lecție



- ✓ Fotorezistorul simte lumina prin rezistență; împărțitorul o transformă în tensiune de citit
- ✓ În montajul nostru: multă lumină = număr mare, întuneric = număr mic
- ✓ Pragul se măsoară, nu se ghicește: citești ambele situații și alegi granița dintre ele
- ✓ Senzor + prag + „dacă” = primul dispozitiv care reacționează singur la lume

LECȚIA 3.4 · 50 DE MINUTE

Temperatura: senzorul TMP36 și citirile cu zgomot

În această lecție construiești un termometru digital: transformi numărul convertorului în volți și volții în grade, apoi înveți primul instrument al oricărui inginer de date: media care netezește zgomotul.

50 MIN · DIFICULTATE ●●○ · CALCUL + COD · AMBELE TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să transformi o citire brută în grade Celsius, prin formula senzorului sau prin calibrare, în funcție de placă
- Să folosești variabile: zone de memorie cu nume, în care programul păstrează valori
- Să netezești citirile cu zgomot folosind media aritmetică

SECȚIUNEA 01 · TEORIE

Din număr în grade, în doi pași

DEJA CUNOȘTI (DIN LECȚIILE 3.2 ȘI 3.3)

Convertorul dă un număr 0–1023, proporțional cu tensiunea de pe pin. Astăzi numărul acela devine grade și afli ce faci când citirile nu sunt la fel de curate ca în simulator.

Senzorul de temperatură TMP36 furnizează pe terminalul din mijloc o tensiune care crește liniar cu temperatura: fiecare grad în plus adaugă exact 0,01 V (10 mV/°C). Numărul citit de placă trebuie deci tradus în două trepte: întâi din număr în volți (proporția convertorului, aplicată invers), apoi din volți în grade (formula din fișa tehnică a senzorului).

Formula lecției: conversia TMP36

volți = citire × tensiunea plăcii ÷ 1023 grade = (volți − 0,5) × 100

De unde vine 0,5? Senzorul este construit să furnizeze exact 0,5 V la 0 °C, pentru a putea indica și temperaturi sub zero fără tensiuni negative (acest decalaj se numește offset). Exemplu complet, la Arduino (5 V): citirea 153 → volți = 153 × 5 ÷ 1023 ≈ 0,75 V → grade = (0,75 − 0,5) × 100 = 25 °C. La Micro:bit, formula este aceeași, dar tensiunea de referință din simulator nu este cea din fișa tehnică, așa că formula nu se aplică direct. Vezi caseta de mai jos.

⚠ Formula este a senzorului real. Simulatorul are altă scară.

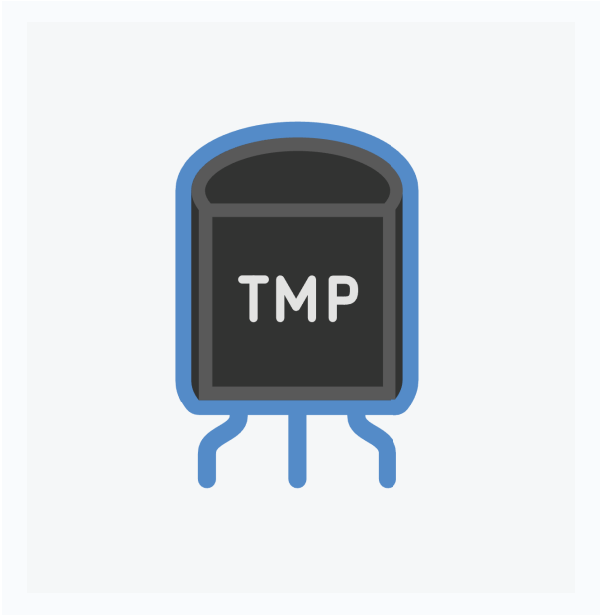
Formula de mai sus este cea din fișa tehnică a senzorului TMP36 și funcționează pe o placă reală. În Tinkercad însă, combinația Micro:bit + TMP36 răspunde altfel: tot liniar (cu cât este mai cald, cu atât numărul este mai mare, în pași egali), dar cu alt punct de plecare și cu altă pantă. Dacă aplici formula ca atare, vei citi în jur de 23 °C acolo unde senzorul arată 11, iar eroarea se schimbă cu temperatura, deci nu este o simplă abatere pe care să o scazi. La rece dă prea mult, la cald prea puțin, iar în jur de 60 °C nimerește din întâmplare: semn sigur că nu este o eroare de reglaj, ci o scară diferită.

Nu este o defecțiune și nu este nimic de reparat în circuit. Este situația pe care o întâlnește orice inginer când senzorul din mână nu se comportă exact ca cel din fișa tehnică: atunci nu îl crezi pe cuvânt, ci îl calibrezi.

SECȚIUNEA 02 · PIESA DE AZI

Piesă nouă: Senzorul de temperatură TMP36

PRIMA APARIȚIE



CE ESTE

O capsulă neagră cu trei terminale; seamănă perfect cu tranzistorul din Lecția 2.3, dar în interior se află un termometru electronic. Terminalele, cu fața plată spre tine: alimentare · semnal · GND.

CE FACE

Furnizează pe terminalul din mijloc o tensiune proporțională cu temperatura: 0,5 V la 0 °C, plus 0,01 V pentru fiecare grad. În simulare, execuți clic pe el și tragi glisorul de temperatură.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ

Senzori înrudiți măsoară temperatura în telefoane, laptopuri, centrale termice și stații meteo: orice aparat care „își cunoaște” temperatura.

DE CE NE TREBUIE AZI

Este primul senzor cu fișă tehnică adevărată: numărul lui devine grade doar prin calcul; exact activitatea de zi cu zi a achiziției de date.

Variabilele și zgomotul

Calculul în două trepte cere păstrarea rezultatelor intermediare; de aceea, astăzi intră oficial în scenă variabila: o zonă de memorie cu nume, în care programul așază o valoare pentru a o folosi mai târziu. Ai întâlnit-o deja în 3.2 (valoare = ...); astăzi folosim trei: citire, volți, grade. Numele bine alese fac programul lizibil ca o rețetă.

Și încă un adevăr din laborator: citirile reale nu stau perfect nemișcate, ci variază cu câteva unități, din cauza unor mici perturbații electrice. Această variație se numește zgomot, iar primul instrument împotriva lui este cunoscut de mult: media aritmetică. Citești de trei ori, aduni, împarți la 3: vârfurile întâmplătoare se anulează reciproc, iar valoarea se stabilizează. Toate aparatele de măsură serioase fac exact acest lucru, doar cu mai multe citiri.

SECȚIUNEA 03 · PRACTICĂ

Termometrul digital

Cablajul, comun: cu fața plată a senzorului spre tine: terminalul stâng → alimentare (3V / 5V), cel din mijloc → pinul analogic (P0 / A0), cel din dreapta → GND. Atenție la ordine: inversat, senzorul se încălzește în realitate; în simulator scapi doar cu valori fără sens.

CIRCUIT DE START

▶ Deschide circuitul de start 3.4-A (Micro:bit)

Placa și senzorul TMP36, cu cele trei fire neconectate. Apasă „Copy and Tinker”.

CIRCUIT DE START

▶ Deschide circuitul de start 3.4-B (Arduino)

Placa și senzorul TMP36, cu cele trei fire neconectate. Apasă „Copy and Tinker”.

TRASEUL A · MICRO:BIT

Semnalul → P0 · calculul și afișarea în Python

1 · Cablează senzorul și pornește termometrul

Trei fire, iar ordinea terminalelor contează. Ține senzorul cu fața plată spre tine și citește de la stânga la dreapta:

- terminalul din stânga → alimentare (3V)
- terminalul din mijloc → P0
- terminalul din dreapta → GND

Cu terminalele inversate, senzorul se încălzește și dă valori eronate. Scrie apoi programul de mai jos, pornește simularea și trage glisorul de temperatură al senzorului: numărul afișat urmărește glisorul.

☐ Termometrul afișează grade care urmăresc glisorul

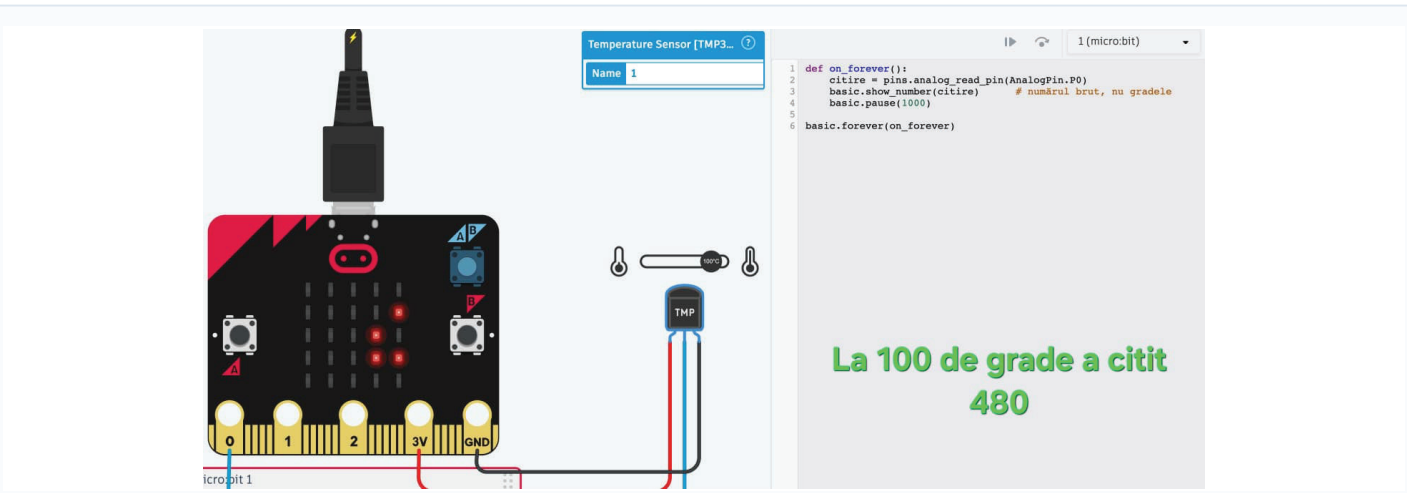
2 · Calibrează: două puncte, o dreaptă

Pentru a traduce numărul în grade nu ai nevoie de fișa tehnică, ci de două măsurători ale tale. Înlocuiește pentru moment rândul de afișare cu unul care arată numărul brut: basic.show_number(citire).

Pornește simularea și pune glisorul senzorului pe 0 °C și notează numărul. Apoi pe 100 °C și notează-l și pe acesta. Două puncte determină o dreaptă, iar dreapta îți dă orice temperatură dintre ele.

Formula este o regulă de trei simplă: scazi numărul de la 0 °C, apoi împarți la cât crește numărul pentru 100 de grade. La noi au rezultat 223 și 480, deci: **grade = (citire − 223) × 100 ÷ 257**. La tine valorile ar putea fi ușor diferite; folosește propriile numere.

☐ Termometrul arată aceleași grade ca glisorul senzorului



TERMOMETRUL · VARIANTA ÎN TEXT

PYTHON · MICRO:BIT

TERMOMETRUL · VARIANTA ÎN TEXT **PYTHON · MICRO:BIT**

```
# Lecția 3.4: termometru digital
# (varianta Tinkercad / MakeCode)

# Cele două numere măsurate de tine, la pasul 2.
# Înlocuiește-le cu ale tale dacă au ieșit altfel.
CITIRE_LA_0 = 223     # numărul citit cu glisorul pe 0 °C
CITIRE_LA_100 = 480   # numărul citit cu glisorul pe 100 °C

def on_forever():
    citire = pins.analog_read_pin(AnalogPin.P0)   # treapta 0: numărul brut
    # treapta 1: cât de sus este numărul față de punctul de zero
    peste_zero = citire - CITIRE_LA_0
    # treapta 2: câte grade înseamnă acest lucru, prin regula de trei
    grade = peste_zero * 100 / (CITIRE_LA_100 - CITIRE_LA_0)
    basic.show_number(Math.round(grade))   # rotunjit la întreg
    basic.pause(1000)

basic.forever(on_forever)
```

În final, senzorul de temperatură va citi datele corect, cu o potențială eroare de 1–2 grade.

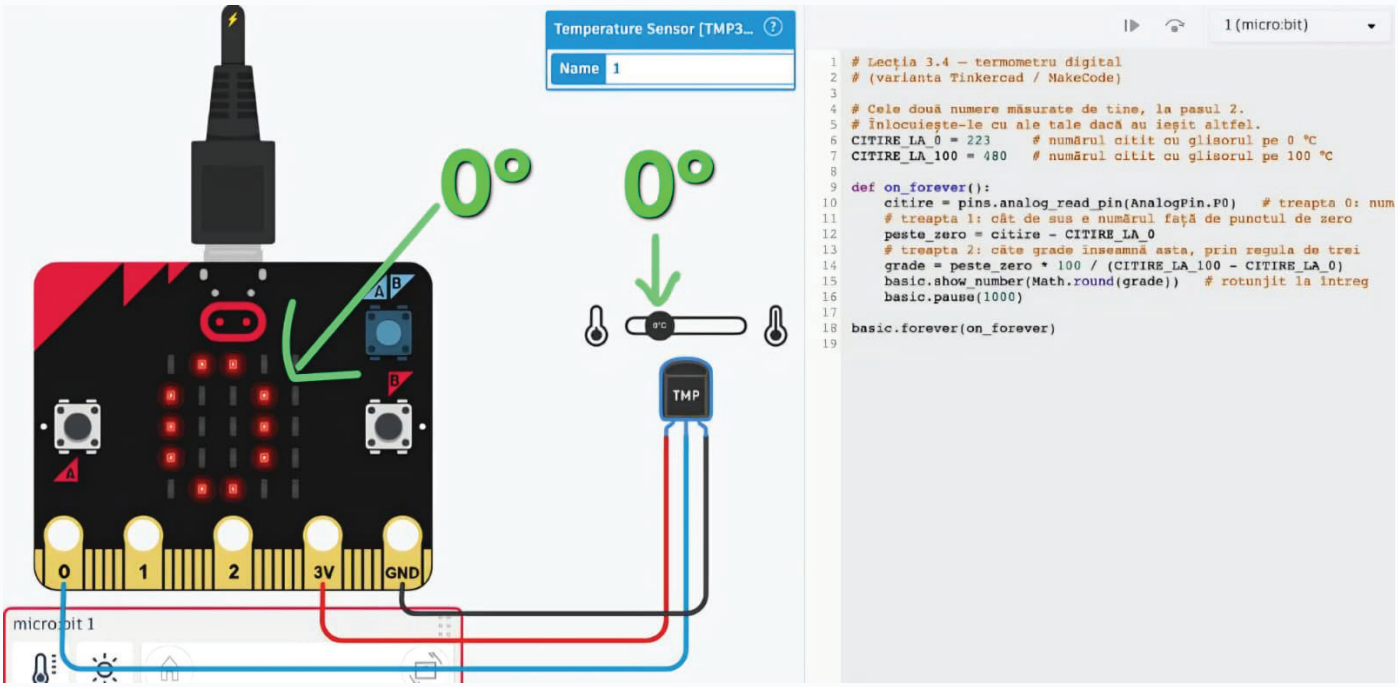


Fig. 1. Termometrul calibrat: gradele de pe afișaj urmăresc glisorul de temperatură al senzorului.

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

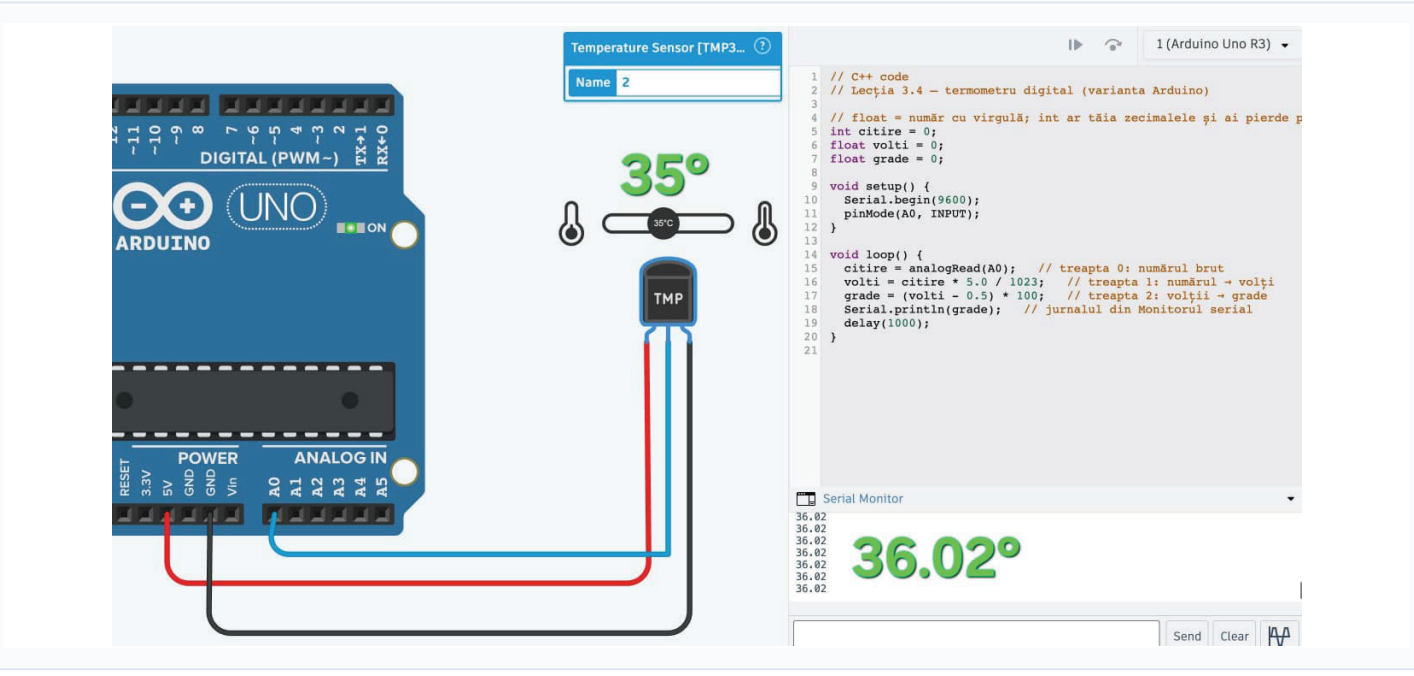
TRASEUL B · ARDUINO UNO

Semnalul → A0 · calculul în C++, jurnalul în Monitorul serial

1 · De ce nu merge în blocuri?

Cablajul pentru Arduino este același ca și pe micro:bit, fiind diferiți doar pinii (5V – Power, A0 – Vout, GND – GND). Editorul de blocuri din Tinkercad compilează totul în C++ real, iar variabilele create din read analog pin sunt automat de tip întreg. Când împarți citire × 5 ÷ 1023 cu toate numerele întregi, rezultatul poate fi doar 0 sau 1, niciodată o valoare intermediară. De aici, grade iese mereu −50 sau 50, oricât ai mișca glisorul senzorului. Nu poți repara acest lucru doar din blocuri: nu există casetă în care să scrii „5.0" în loc de „5". Trecem direct în editorul de text, unde declari variabilele ca float, adică numere cu virgulă, de la bun început.

☐ Gradele din jurnal urmăresc glisorul



Cablajul pentru Arduino este același ca și pe micro:bit, fiind diferiți doar pinii (5V – Power, A0 – Vout, GND – GND).

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ C++ · ARDUINO

```
// C++ code
// Lecția 3.4: termometru digital (varianta Arduino)

// float = număr cu virgulă; int ar tăia zecimalele și ai pierde precizia
int citire = 0;
float volti = 0;
float grade = 0;

void setup() {
  Serial.begin(9600);
  pinMode(A0, INPUT);
}

void loop() {
  citire = analogRead(A0);   // treapta 0: numărul brut
  volti = citire * 5.0 / 1023;   // treapta 1: numărul → volți
  grade = (volti - 0.5) * 100;   // treapta 2: volții → grade
  Serial.println(grade);   // jurnalul din Monitorul serial
  delay(1000);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

2 · Netezește zgomotul cu media

Înlocuiește rândul citirii cu media a trei citiri: **citire = (analogRead(A0) + analogRead(A0) + analogRead(A0)) / 3;** Restul programului nu se schimbă deloc: media intră în aceleași două trepte. În simulator numărul nu se va schimba, deoarece senzorii de aici dau valori perfect curate. Pe un senzor real însă, unde citirile variază cu câteva unități, exact acest rând face diferența dintre un termometru care sare și unul care stă liniștit.

☐ Am înlocuit citirea cu media a trei citiri

Provocare: calibrează și Arduino-ul

Pe traseul B, codul folosește formula din fișa tehnică: volți = citire × 5 ÷ 1023, apoi grade = (volți − 0,5) × 100. Verific-o: pune glisorul pe 0 °C și pe 100 °C și compară gradele afișate cu cele cerute.

Dacă se potrivesc, ai aflat ceva important: modelul de Arduino din simulator respectă fișa tehnică, iar Micro:bit-ul nu. Dacă nu se potrivesc, aplică exact metoda de la pasul 2: notează cele două numere brute și înlocuiește formula cu regula de trei. Același instrument, altă placă.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT
Calculul în Python este curat ca pe caiet: trei rânduri, trei variabile; matematica se vede direct.

ARDUINO UNO
Codul C++ este ușor de implementat; Monitorul serial rămâne cel mai bun loc de urmărit un termometru.

Concluzia: formulele lungi se scriu mai frumos în text, datele se urmăresc mai bine în jurnal. Combinația câștigătoare a inginerilor: cod-text + monitor de date.

SECȚIUNEA 04 · DEPANARE

⚠ Gradele ies aiurea?

1) Grade uriașe sau negative fără sens → terminalele senzorului sunt inversate; verifică ordinea cu fața plată spre tine. 2) Mereu aceeași valoare → semnalul nu este în pinul analogic. 3) Grade mult prea mari pe micro:bit → nu ai înlocuit cele două numere de calibrare cu ale tale. 4) Python: împărțirea și înmulțirea folosesc * și /, nu x și :.

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă (6 minute, toată clasa)

Profesorul arată o imagine (sau un obiect) și cere o estimare numerică: „câte grade sunt acum în clasă?”, „câte bomboane încap în borcan?”. Fiecare elev scrie estimarea lui pe foaie; un elev-secretar calculează media tuturor estimărilor. Apoi vine adevărul (termometrul clasei, numărătoarea). Aproape întotdeauna, media clasei întrece majoritatea estimărilor individuale: greșelile în sus și în jos se anulează reciproc. Exact de aceea termometrul tău citește de trei ori și face media: fiecare citire este o „părere” cu zgomot, iar media este clasa întreagă.

Varianta Kahoot: estimări numerice cu verificare, plus calcule de medie contra cronometru; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 3.4-J: trei seturi de citiri cu zgomot; calculează media fiecăruia și compară cu valoarea reală dată
Varianta fără ecrane: chiar jocul principal; necesită foi, un secretar și ceva de estimat

Provocare: termostatul

Unește lecțiile 3.3 și 3.4: alege un prag de temperatură (de exemplu 28 °C) și aprinde LED-ul de „prea cald” când gradele îl depășesc: schița oricărui termostat de centrală. Bonus: în Modulul 4, LED-ul devine ventilator adevărat.

3 întrebări rapide

1. Care sunt cele două trepte prin care citirea brută devine grade Celsius?
2. Ce este o variabilă și de ce numele ei contează?
3. Ce este zgomotul citirilor și cum îl liniștește media?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

TMP36: senzorul de temperatură: 0,5 V la 0 °C, plus 0,01 V pe grad

fișă tehnică: documentul oficial al unei componente: pinii, formulele, limitele ei

variabilă: zonă de memorie cu nume, în care programul păstrează o valoare

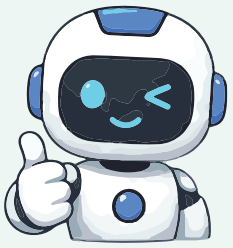
zgomot: variația măruntă și întâmplătoare a citirilor reale

medie aritmetică: suma citirilor împărțită la numărul lor; netezește zgomotul

float: în C++, tipul numerelor cu virgulă (în vitrina Arduino)

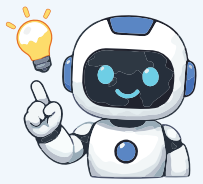
Ce trebuie să știi după această lecție

- ✓ Citirea devine grade în două trepte; pe Arduino prin formula din fișa tehnică, pe Micro:bit prin calibrare în două puncte
- ✓ Când senzorul nu se poartă ca în carte, îl măsori în două puncte cunoscute și tragi dreapta prin ele
- ✓ Variabilele sunt zone de memorie cu nume; trei nume bine alese fac formula lizibilă ca o rețetă
- ✓ Citirile reale variază; media mai multor citiri le stabilizează: prima regulă a achiziției de date



LECȚIA 3.5 · 50 DE MINUTE

Distanța și mișcarea: senzorul ultrasonic și senzorul PIR



Doi senzori care văd fără să atingă: unul măsoară distanța cu ecoul sunetului, celălalt simte mișcarea după căldura corpului. Cu ei construiești senzorul de parcare și alarma de mișcare.

50 MIN · DIFICULTATE ●●● · DOI SENZORI NOI · AMBELE TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să explici principiul ecoului și formula distanței
- Să construiești o alarmă de mișcare cu senzorul PIR
- Să construiești senzorul de parcare: LED-uri care se aprind pe măsură ce obiectul se apropie

SECȚIUNEA 01 · TEORIE

Măsurarea cu ecou

DEJA CUNOȘTI (DIN LECȚIILE 3.1 ȘI 3.4)

Citirea digitală răspunde cu 1 sau 0: exact limbajul senzorului PIR de astăzi. Iar temperatura din 3.4 apare și aici, într-un detaliu-surpriză la finalul teoriei.

Strigă într-o vale și numără secundele până se întoarce ecoul: cu cât muntele este mai departe, cu atât aștepți mai mult. Senzorul ultrasonic face exact acest lucru, de zeci de ori pe secundă, cu un sunet cu frecvență prea înaltă pentru urechea umană (ultrasunet, peste 20 kHz): un „difuzor” trimite impulsul, un „microfon” prinde ecoul, iar electronica măsoară timpul dus-întors.

Formula lecției: distanța din timp

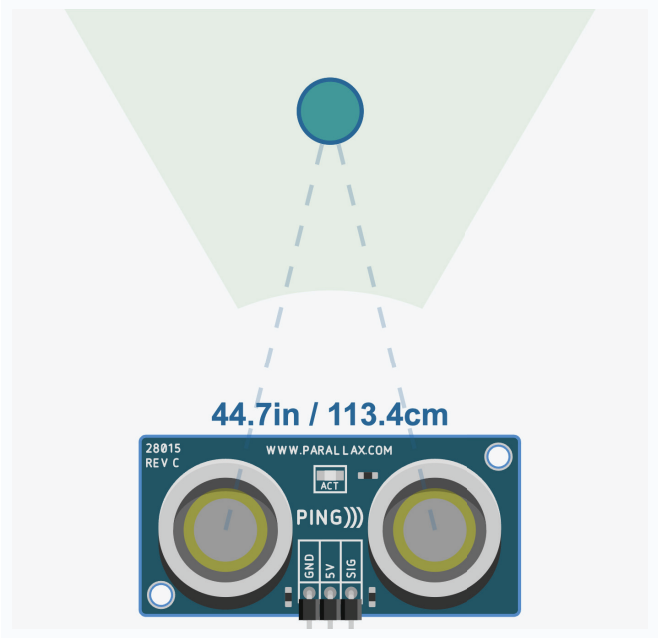
distanța = viteza sunetului × timpul ÷ 2

Împărțim la 2 deoarece timpul măsurat este dus-întors, iar pe noi ne interesează doar dusul. Viteza sunetului în aer este de circa 343 m/s (la 20 °C). Exemplu: ecoul revine după 0,002 secunde → distanța = 343 × 0,002 ÷ 2 ≈ 0,34 m = 34 cm. Detaliul-surpriză promis: viteza sunetului crește ușor cu temperatura (cu aproximativ 0,6 m/s pentru fiecare grad); de aceea, senzorii de parcare foarte precișii țin cont și de temperatura de afară, exact cea din Lecția 3.4.

SECȚIUNEA 02 · PIESELE DE AZI

Piesă nouă: Senzorul ultrasonic de distanță

PRIMA APARIȚIE



CE ESTE

O plăcuță cu doi cilindri metalici asemenea unei perechi de ochi: emițătorul (difuzorul de ultrasunete) și receptorul (microfonul ecoului). Pinii lui: alimentare, GND și semnalul; în Tinkercad, aceeași linie (SIG) trimite impulsul și ascultă ecoul, iar la modelele cu patru terminale, declanșarea (trigger) și ecoul (echo) au fiecare pinul lor.

CE FACE

Trimite un impuls de ultrasunete, cronometrează ecoul și raportează distanța până la primul obiect din față, de la câțiva centimetri până la câțiva metri. În simulare, obiectul-țintă apare lângă senzor și îl tragi cu mouse-ul mai aproape sau mai departe.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ

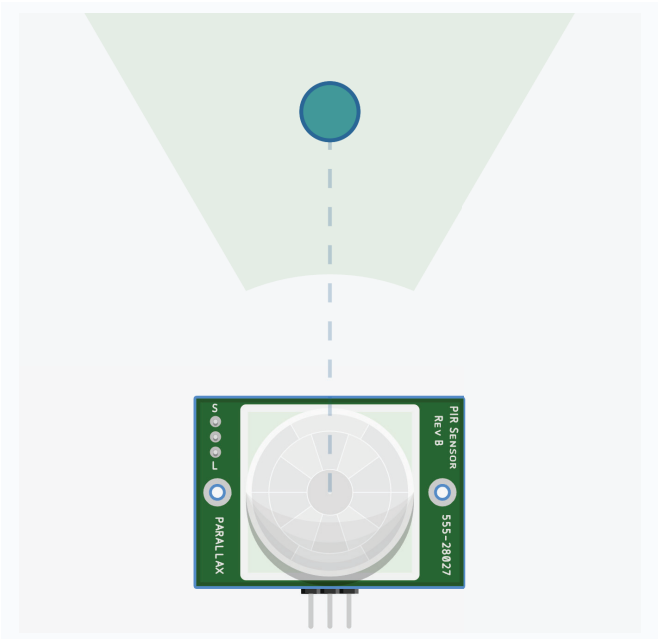
Senzorii de parcare ai mașinilor (piuitul tot mai des), ușile care se deschid singure, roboții care ocolesc obstacole.

DE CE NE TREBUIE AZI

Este primul senzor care măsoară spațiul din jur: pasul de la „cât de cald este” la „ce se află în fața mea”.

Piesă nouă: Senzorul de mișcare PIR

PRIMA APARIȚIE



CE ESTE

O plăcuță cu o lentilă albă, bombată, ca o jumătate de minge de ping-pong. Sub ea, un detector de radiație termică; PIR vine din limba engleză, Passive InfraRed, adică „infraroșu pasiv”: nu emite nimic, doar recepționează.

CE FACE

Simte schimbarea căldurii din fața lui: un om care trece îi mută „petele calde” din câmpul vizual, iar senzorul ridică semnalul pe 1 pentru câteva clipe. Răspunsul lui este digital: mișcare (1) sau liniște (0); se citește exact ca un buton.

UNDE ÎL ÎNTÂLNEȘTI ÎN VIAȚA REALĂ

Lumina de pe casa scării care se aprinde când urci, alarmele de locuință, uscătoarele de mâini automate.

DE CE NE TREBUIE AZI

Este dovada că Lecția 3.1 ajunge departe: un senzor „serios” de securitate se citește cu aceeași instrucțiune ca un buton.

SECȚIUNEA 03 · PRACTICĂ

Alarma de mișcare, apoi senzorul de parcare

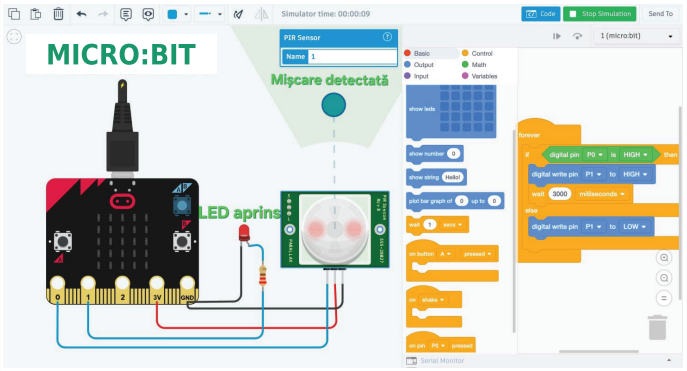
PARTEA 1 · ALARMA DE MIȘCARE · PE AMBELE PLĂCI

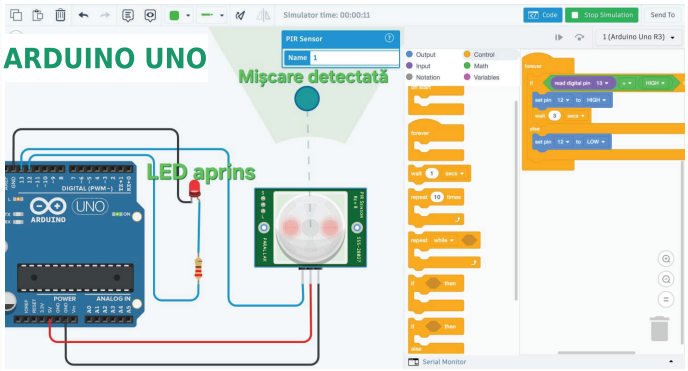
Alarma de mișcare cu PIR: semnalul → P0 / pinul 13 · LED-ul de alarmă pe P1 / pinul 12

1 · Cablează PIR-ul și refolosește decizia din 3.1

PIR: alimentare → 3V / 5V, GND → GND, semnalul → pinul de intrare. Programul este cel de la buton, cu aceeași structură; se schimbă doar pinii: dacă semnalul PIR este 1, LED-ul de alarmă este aprins; altfel, stins. Am adăugat și o pauză de 3 secunde pe ramura „atunci", ca alarma să rămână aprinsă o clipă după ce mișcarea a trecut. Pornește simularea și execută clic lângă senzor: apare punctul de „mișcare” pe care îl plimbi prin fața lentilei.

☐ Alarma prinde mișcarea din simulare





⚠ De ce nu funcționează PIR-ul real doar cu micro:bit?

În Tinkercad, senzorul PIR funcționează perfect alimentat direct din pinul de 3 V al plăcii. În realitate, lucrurile stau puțin altfel: modulul PIR standard (modelul HC-SR501) are pe el un mic stabilizator de tensiune care „consumă” o parte din tensiune. Din acest motiv, el cere minimum 4,5–5 V pentru a funcționa.

Dacă vei construi acest circuit acasă cu componente fizice, exact ca pe planșă, senzorul tău va rămâne complet mut. Pentru a-l face să funcționeze în lumea reală, ai nevoie fie de o alimentare separată de 5 V pentru el, combinată cu un împărțitor de tensiune pe firul de semnal (ieșirea lui de 5 V trebuie coborâtă la 3,3 V înainte de pinul plăcii), fie de o versiune de modul PIR fabricată special pentru 3,3 V.

PARTEA 2 · SENZORUL DE PARCARE · SE LUCREAZĂ PE ARDUINO, ORICARE AR FI TRASEUL TĂU

Senzorul de parcare: ultrasonic + 3 LED-uri pe praguri de distanță

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT
PIR funcționează identic aici. Pentru ultrasonic însă, Python-ul cere comenzi de cronometrare avansate (pe placa reală se poate, dar în simulator nu); de aceea, partea a doua se lucrează pe Arduino.

ARDUINO UNO
Blocul dedicat de ultrasonic face toată munca: pinii, cronometrul, formula. De aceea partea 2 se lucrează aici, indiferent de traseul tău obișnuit.

Concluzia: primul caz din manual în care o placă câștigă detașat o probă: la senzorul ultrasonic, Arduino este alegerea simplă. Alegerea plăcii în funcție de proiect: exact acest lucru exersezi.

CIRCUIT DE START

▶ Deschide circuitul de start 3.5 (senzor de parcare)

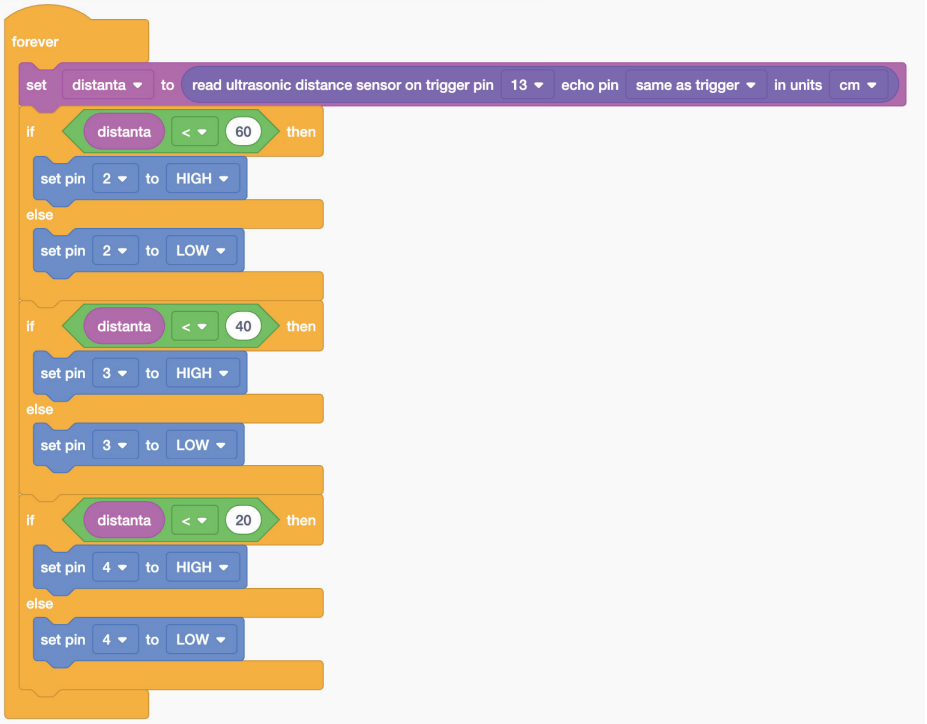
Arduino, senzorul ultrasonic și 3 LED-uri cu rezistorii lor, gata așezate. Apasă „Copy and Tinker”.

2 · Citește distanța dintr-un singur bloc

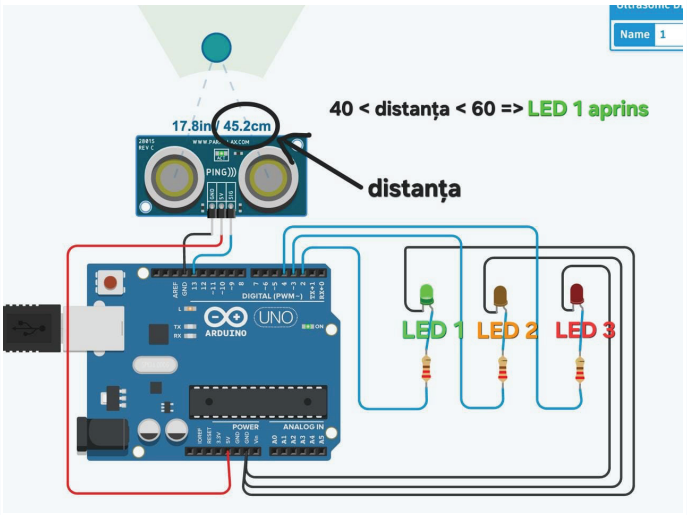
Semnalul de la senzorul ultrasonic (SIG) este conectat la pinul digital 13, împreună cu alimentarea la 5V și GND. LED-urile sunt legate astfel: LED 1 (stânga) la pinul 2, LED 2 (mijloc) la pinul 3, iar LED 3 (dreapta) la pinul 4.

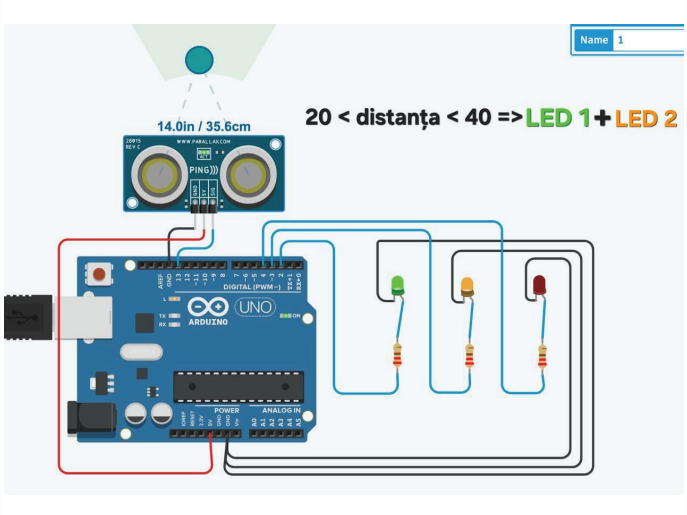
Trei decizii independente, fiecare cu ramura ei de stingere: dacă distanța < 60 → primul LED aprins, altfel stins; dacă < 40 → al doilea aprins, altfel stins; dacă < 20 → al treilea aprins, altfel stins. Sub 20 cm sunt aprinse toate trei deodată; peste 60, toate stinse. Trei praguri, trei „dacă... altfel": decizia din Lecția 3.3, repetată de trei ori.

☐ LED-urile se aprind pe rând când ținta se apropie și se sting când o îndepărtezi

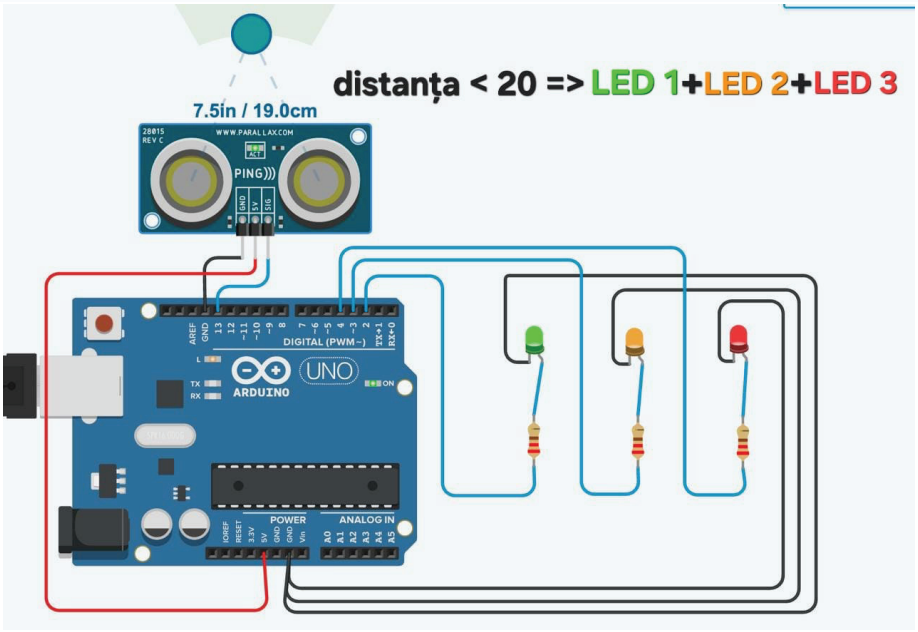


Trei praguri, trei decizii independente





La distanțe sub 60 cm se aprinde primul LED; sub 40 cm se adaugă al doilea. Fiecare prag este evaluat separat, astfel încât LED-urile se cumulează pe măsură ce obiectul se apropie.



Sub 20 cm sunt aprinse toate trei LED-urile: distanța a trecut de toate cele trei praguri.

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ · PARTEA 1 C++ · ARDUINO

```
// C++ code
// Lecția 3.5: senzorul de parcare (varianta Arduino)

// Variabilă în care vom stoca distanța calculată, în centimetri
int distanta = 0;

// Funcție ajutătoare care comandă senzorul ultrasonic să emită și să asculte sunetul
long readUltrasonicDistance(int triggerPin, int echoPin)
{
  // Setăm pinul ca ieșire pentru a putea genera semnalul sonor
  pinMode(triggerPin, OUTPUT);

  // Ne asigurăm că pinul este pe LOW (oprit) înainte de a emite
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);

  // Generăm "strigătul" senzorului: un scurt impuls de 10 microsecunde
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);

  // Schimbăm pinul pe intrare pentru a "asculta" ecoul
  pinMode(echoPin, INPUT);

  // Măsurăm cât timp a trecut până când ecoul s-a întors la noi (în microsecunde)
  return pulseIn(echoPin, HIGH);
}
```

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ · PARTEA 2 C++ · ARDUINO

```
void setup() {
  // Configurăm pinii la care sunt conectate LED-urile ca ieșiri (OUTPUT)
  pinMode(2, OUTPUT);   // LED-ul de departe (ex. verde)
  pinMode(3, OUTPUT);   // LED-ul de mijloc (ex. galben)
  pinMode(4, OUTPUT);   // LED-ul de aproape (ex. roșu)
}

void loop() {
  // Măsurăm timpul de zbor al sunetului pe pinul 13 și calculăm distanța.
  // 0.01723 transformă timpul total (dus-întors) direct în centimetri la dus.
  distanta = 0.01723 * readUltrasonicDistance(13, 13);

  // Evaluăm distanța și aprindem sau stingem LED-urile corespunzătoare:

  // 1. Pragul de 60 cm (cel mai îndepărtat)
  // Dacă obiectul e la mai puțin de 60 cm, aprindem primul LED, altfel îl stingem
  if (distanta < 60) { digitalWrite(2, HIGH); } else { digitalWrite(2, LOW); }

  // 2. Pragul de 40 cm (obiectul se apropie)
  // Dacă e la mai puțin de 40 cm, se aprinde și al doilea LED
  if (distanta < 40) { digitalWrite(3, HIGH); } else { digitalWrite(3, LOW); }

  // 3. Pragul de 20 cm (pericol de coliziune)
  // Dacă obiectul a ajuns la sub 20 cm, se aprinde și ultimul LED
  if (distanta < 20) { digitalWrite(4, HIGH); } else { digitalWrite(4, LOW); }
  delay(100); // Pauză între citiri
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

SECȚIUNEA 04 · DEPANARE

⚠ Senzorii tac?

1) PIR pare „blocat pe 1” câteva secunde după mișcare? Este normal: senzorul menține semnalul ridicat o vreme; așa este construit. 2) Distanța nu se schimbă? Trage chiar obiectul-țintă al senzorului, nu senzorul însuși. Verifică și numărul de terminale: Tinkercad așază senzorul implicit în varianta cu trei terminale (semnal unic); pentru trigger și echo separate, schimbă-l din caseta de proprietăți. 3) LED-urile rămân aprinse după ce îndepărtezi ținta? Îți lipsesc ramurile „altfel": fiecare „dacă" are nevoie și de instrucțiunea care stinge. 4) Blocul de ultrasonic cere pinii corecți; compară cu captura.

SECȚIUNEA 05 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, cu mișcare)

Un elev voluntar, legat la ochi cu o eșarfă, este mașina; peretele clasei este obstacolul. Restul clasei este senzorul: pe măsură ce „mașina” se apropie de perete, clasa bate din palme tot mai des (rar la 3 metri, des la 1 metru, continuu sub jumătate de metru). Misiunea „mașinii”: să oprească la o palmă de perete, doar după ritm. Se schimbă voluntarii, se măsoară cine a oprit cel mai aproape fără să atingă. Clasa tocmai a jucat exact algoritmul senzorului de parcare: distanță → praguri → semnal tot mai insistent.

Varianta Kahoot: calcule cu formula ecoului, plus situații „ce senzor detectează acest lucru?”; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 3.5-J: trei timpi de ecou dați; calculează distanțele; plus schema senzorului de parcare de completat cu praguri
Varianta fără ecrane: chiar jocul principal; necesită doar o eșarfă și palme

Provocare: parcarea sonoră

Adaugă componenta „Piezo” (difuzorul mic din dulapul cu piese) și fă senzorul de parcare să și piuie, nu doar să lumineze: sub 20 cm, sunet continuu. Exact ca în automobil. Indiciu: blocul de ton are nevoie de pin și de o frecvență; încearcă 440 Hz.

3 întrebări rapide

1. De ce se împarte la 2 în formula distanței cu ecou?
2. Prin ce se deosebește răspunsul senzorului PIR de cel al senzorului ultrasonic?
3. Ce legătură are temperatura din Lecția 3.4 cu precizia senzorului ultrasonic?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

ultrasunet: sunet cu frecvență peste 20 kHz, inaudibil pentru urechea umană; instrumentul senzorului de distanță

ecou: unda care se întoarce după ce se reflectă dintr-un obstacol

emițător / receptor: cei doi „ochi” ai senzorului: difuzorul și microfonul ecoului

PIR: senzor pasiv de infraroșu: simte schimbarea căldurii = mișcare

semnal de moment: semnalul PIR ținut pe 1 câteva clipe după detecție

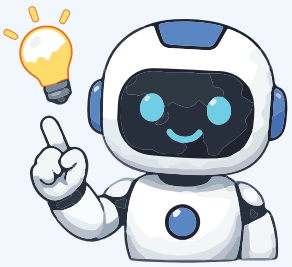
piezo: difuzor piezoelectric; transformă semnalele electrice în sunet

Ce trebuie să știi după această lecție

- ✓ Distanța = viteza sunetului × timpul ÷ 2; ecoul este dus-întors
- ✓ Ultrasonicul răspunde cu un număr (distanța în centimetri) obținut din durata unui impuls, exact cum servomotorul din 2.6 primea unghiul tot ca durată de impuls; PIR-ul răspunde cu 1/0, ca un buton
- ✓ Pragurile în trepte transformă o distanță într-un semnal tot mai insistent: senzorul de parcare
- ✓ Alegerea plăcii în funcție de proiect: la ultrasonic, blocul dedicat al Arduino câștigă proba

LECȚIA 3.6 · 50 DE MINUTE

Senzorii roboților adevărați și fuziunea senzorilor



Ultimul pas al modulului: atlasul senzorilor purtați de mașinile autonome, drone și aspiratoarele robot, și prima ta decizie luată din doi senzori deodată.

50 MIN · DIFICULTATE ●●● · ATLAS + CONSTRUCȚIE CU DOI SENZORI

CE VEI ȘTI LA FINALUL LECȚIEI

- Să recunoști marile familii de senzori și ce măsoară fiecare
- Să explici principiul LiDAR-ului și al camerelor de adâncime prin ce ai exersat deja
- Să scrii o condiție cu operatorul ȘI: doi senzori, o singură decizie

SECȚIUNEA 01 · TEORIE

Atlasul senzorilor

DEJA CUNOȘTI (DIN LECȚIILE 3.1–3.5)
Buton, potențiometru, fotorezistor, TMP36, ultrasonic, PIR: șase senzori exersați cu mâna ta. Toți urmează același lanț: fenomen → semnal → număr → decizie. Atlasul de astăzi arată că și cei mai scumpi senzori din lume urmează exact același lanț.

Senzorii roboticii se împart în câteva mari familii, după ce măsoară: proximitate (ce e în jur, fără atingere), vedere (imagini), forță și atingere (contactul), mișcare (cum se mișcă robotul însuși) și poziție și navigație (unde se află). Pe primele două le-ai deschis deja cu ultrasonicul și fotorezistorul; iată vârfurile fiecărei familii; pe acestea nu le putem simula în Tinkercad, dar principiile lor le-ai exersat deja:

LiDAR

Cameră de adâncime

Accelerometru

Hartă de puncte

SENZORUL	CE FACE	UNDE ÎL VEZI
LiDAR	Fratele cu lumină al senzorului ultrasonic: trimite impulsuri laser, cronometrează ecoul și repetă măsurarea de sute de mii de ori pe secundă, rotindu-se; rezultatul este o hartă a spațiului din jur (denumirea vine din limba engleză: Light Detection and Ranging).	Mașinile autonome, aspiratoarele robot cu turelă, dronele de cartografiere
Camera	Ochiul robotului: transformă lumina în imagine, iar programele de vedere artificială recunosc în ea obiecte, fețe, marcaje de drum.	Telefoane, mașini (camerele de mers înapoi și de bandă), brațe robotice din fabrici
Camera de adâncime	Vede și cât de departe este fiecare punct din imagine, cu două „ochiuri” care compară imaginile (precum ochii tăi, prin vedere stereoscopică) sau cronometrând lumina, ca un LiDAR pe toată imaginea.	Deblocarea telefonului cu fața, consolele cu detectarea mișcării, roboții care apucă obiecte
Accelerometrul	Simte accelerațiile și înclinarea; el știe încotro este „jos”.	Telefonul care își rotește ecranul, airbag-urile, brățările de pași
Giroscopul	Simte rotirea și o măsoară precis; împreună cu accelerometrul ține echilibrul.	Dronele care stau nemișcate în aer, stabilizatoarele camerelor, trotinetele electrice
Encoderul	Numără rotațiile axului unui motor; de aici robotul știe cât a parcurs fiecare roată.	Roțile roboților, brațele robotice, imprimantele
GPS-ul	Ascultă sateliții și calculează poziția pe glob, cu precizie de câțiva metri.	Navigația mașinii, telefonul, dronele de exterior

Fig. 1. Vârfurile familiilor de senzori. Fiecare urmează lanțul pe care îl cunoști: fenomen → semnal → număr → decizie.

Fuziunea senzorilor și harta din mers

Un singur senzor greșește ușor: camera orbește în întuneric, GPS-ul se pierde în tunel. De aceea, roboții serioși combină mai mulți senzori într-o singură concluzie; tehnica se numește **fuziunea senzorilor**. Vârful ei e **SLAM** (din limba engleză: Simultaneous Localization and Mapping, adică localizare și cartografiere simultană): robotul își construiește harta încăperii chiar în timp ce se mișcă prin ea și se poziționează pe propria hartă. Diferența se vede cu ochiul liber la aspiratoarele robot: cel fără hartă curăță haotic, în zigzag la întâmplare; cel cu SLAM merge în linii ordonate și știe unde a fost. Astăzi faci primul tău pas de fuziune, cu doi senzori pe care îi ai deja.

Operatorul ȘI: doi senzori, o condiție

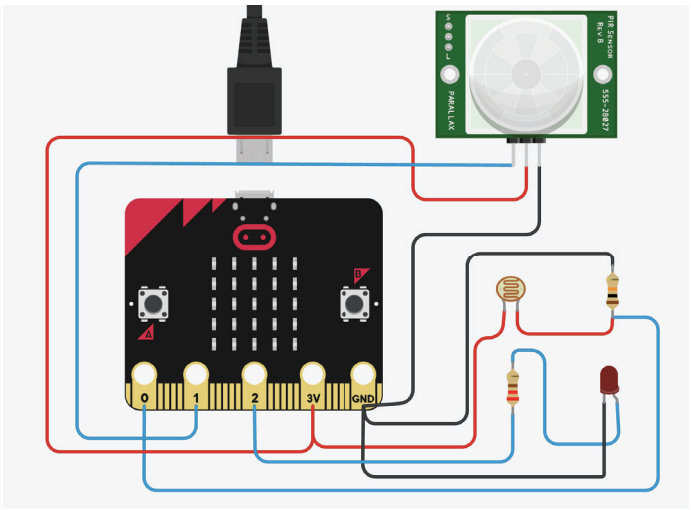
Instrumentul de programare al fuziunii este operatorul logic **ȘI** (and în Python): leagă două condiții într-una singură, adevărată doar când amândouă sunt adevărate. „DACĂ este întuneric ȘI este mișcare, ATUNCI aprinde”: becul de curte perfect, care nu funcționează ziua degeaba și nu se aprinde noaptea fără motiv, ci exact când trece cineva pe întuneric. Există și fratele lui, operatorul SAU (adevărat când cel puțin una este adevărată); îl vei folosi în Modulul 4.

SECȚIUNEA 02 · PRACTICĂ

Becul de curte: întuneric ȘI mișcare

Pe planșă se întâlnesc două lecții: împărțitorul cu fotorezistor din 3.3 (pe P0 / A0) și senzorul PIR din 3.5 (pe P1 / pinul 2). LED-ul-bec, cu rezistorul lui de protecție, pe P2 / pinul 13.

Cablaj pentru Micro:bit



Cablaj pentru Arduino

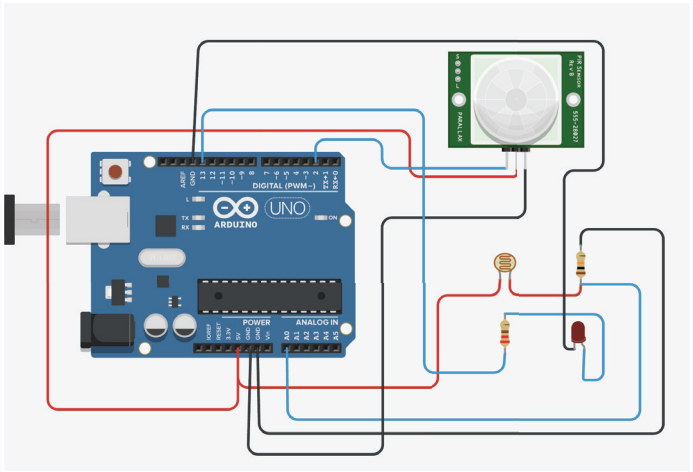


Fig. 2. Două trasee de semnal separate, care se întâlnesc abia în program, în condiția legată cu ȘI.

CIRCUIT DE START

▶ Deschide circuitul de start 3.6-A (Micro:bit)

CIRCUIT DE START

▶ Deschide circuitul de start 3.6-B (Arduino)

Fotorezistorul cu împărțitorul, PIR-ul și LED-ul, gata așezate, neconectate. Apasă „Copy and Tinker”.

1 · Cablează cei doi senzori și becul

LUMINA · ÎMPĂRȚITORUL DIN 3.3

- alimentare (3V / 5V) → un terminal al fotorezistorului
- celălalt terminal al fotorezistorului → rezistorul de 10 kΩ ← punctul de mijloc
- punctul de mijloc → P0 / A0
- celălalt terminal al rezistorului de 10 kΩ → GND

MIȘCAREA · PIR-UL DIN 3.5

- alimentare (3V / 5V) → alimentarea PIR-ului
- GND → GND-ul PIR-ului
- semnalul PIR-ului → P1 / pinul 2

BECUL

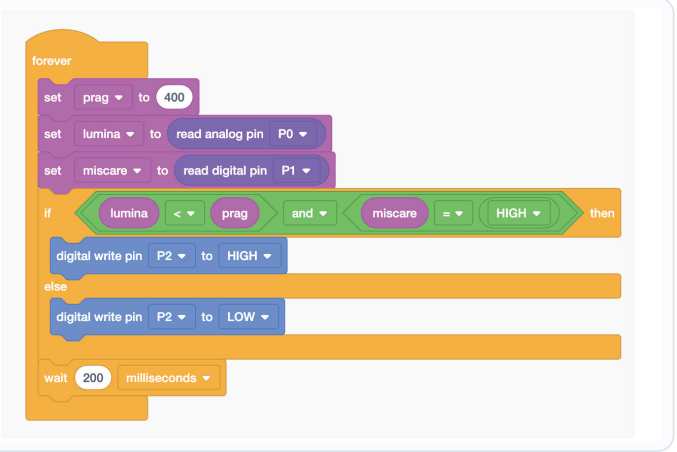
- P2 / pinul 13 → rezistor de protecție → LED → GND

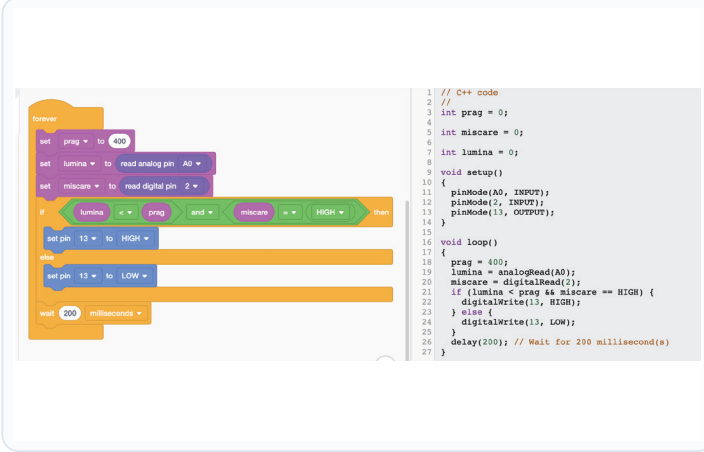
Pinii de alimentare și de GND ai plăcii primesc mai multe fire; este normal: toate componentele se alimentează din aceeași sursă și se întorc pe aceeași masă. Verifică fiecare senzor separat, cu programele din 3.3 și 3.5, înainte să scrii condiția dublă.

☐ Fiecare senzor răspunde corect luat separat

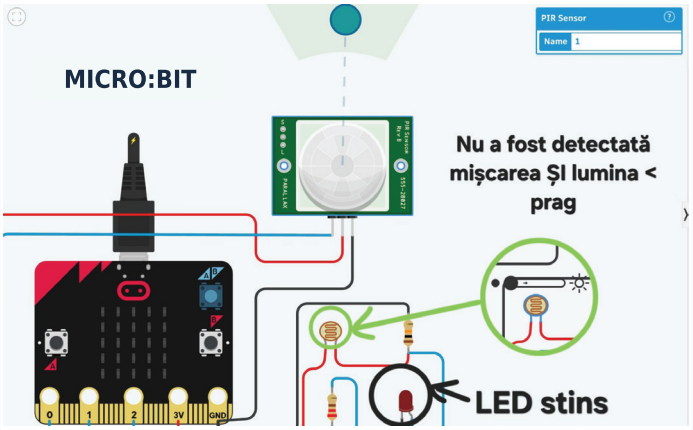
2 · Scrie condiția dublă

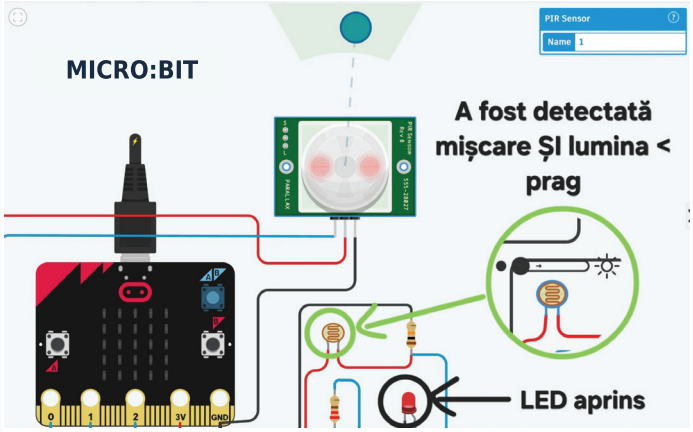
În blocuri, condiția din if devine o pereche legată cu blocul and: (citirea de lumină < PRAG) and (semnalul PIR = 1). În Python: mai jos. Testează toate cele patru combinații trăgând glisorul de lumină și plimbând punctul de mișcare: becul trebuie să se aprindă într-una singură.



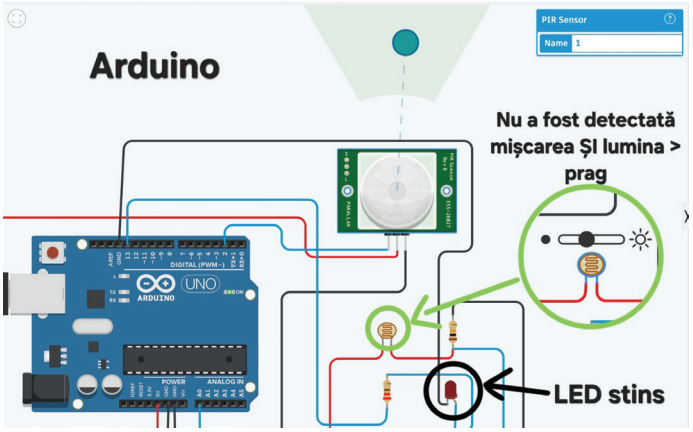


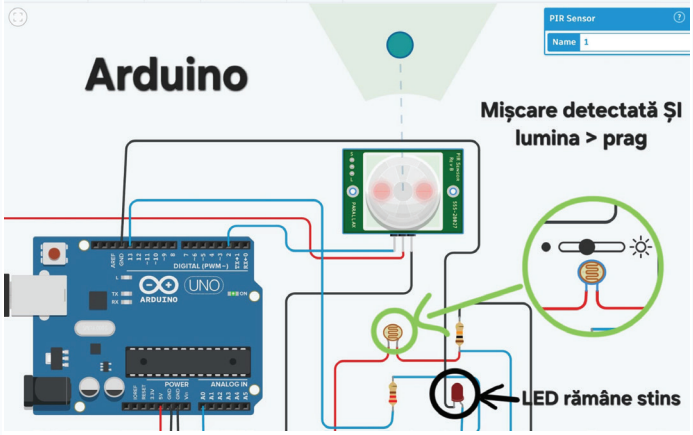
Rezultat: toate cele patru combinații





Pe Micro:bit va fi demonstrată situația în care lumina < PRAG ȘI a fost/nu a fost detectată mișcarea.





Pe Arduino va fi demonstrată situația în care lumină > PRAG ȘI a fost/nu a fost detectată mișcarea.
☐ Becul se aprinde doar la întuneric + mișcare

TRASEUL A · MICRO:BIT

Lumina → P0 · mișcarea → P1 · becul pe P2

BECUL DE CURTE · VARIANTA ÎN TEXT

PYTHON · MICRO:BIT

```
# Lecția 3.6: doi senzori, o decizie
# (varianta Tinkercad / MakeCode)

PRAG = 400   # granița de întuneric, din Lecția 3.3

def on_forever():
    lumina = pins.analog_read_pin(AnalogPin.P0)   # senzorul 1: câtă lumină este
    miscare = pins.digital_read_pin(DigitalPin.P1)   # senzorul 2: este mișcare? (1/0)
    if lumina < PRAG and miscare == 1:   # ambele condiții, legate cu ȘI
        pins.digital_write_pin(DigitalPin.P2, 1)   # atunci: becul aprins
    else:
        pins.digital_write_pin(DigitalPin.P2, 0)   # altfel: stins
    basic.pause(200)

basic.forever(on_forever)
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Lumina → A0 · mișcarea → pinul 2 · becul pe pinul 13

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ — C++ · ARDUINO

```
// C++ code
// Lecția 3.6: doi senzori, o decizie (varianta Arduino)

int PRAG = 400;
int lumina = 0;
int miscare = 0;

void setup() {
  pinMode(A0, INPUT);
  pinMode(2, INPUT);
  pinMode(13, OUTPUT);
}

void loop() {
  lumina = analogRead(A0);   // senzorul 1: câtă lumină este
  miscare = digitalRead(2);   // senzorul 2: este mișcare? (1/0)
  // && este numele operatorului ȘI în limbajul Arduino
  if (lumina < PRAG && miscare == HIGH) {
    digitalWrite(13, HIGH);   // atunci: becul aprins
  } else {
    digitalWrite(13, LOW);   // altfel: stins
  }
  delay(200);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT
Condiția dublă în Python este o propoziție: if lumina < PRAG and miscare == 1; se citește exact ca în limba română.

ARDUINO UNO
Blocul „and” cuprinde vizibil cele două condiții, iar în vitrina C++ apare &&, numele operatorului ȘI în limbajul Arduino.

Concluzia: and, bloc sau &&: trei forme, aceeași logică. Fuziunea senzorilor începe cu un singur cuvânt: ȘI.

SECȚIUNEA 03 · DEPANARE

⚠ Becul nu ascultă de amândoi senzorii?

1) Se aprinde și ziua → condiția de lumină lipsește sau pragul este prea sus. 2) Nu se aprinde niciodată → verifică fiecare senzor separat, cu programele din 3.3 și 3.5, înainte de a-i judeca împreună: regula de aur a depanării cu mai multe componente. 3) În Python, egalitatea din condiții se scrie cu == (două semne), nu cu = (acela atribuie o valoare variabilei).

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, pe echipe)

Fiecare echipă trage un cartonaș cu o misiune și are un minut să aleagă senzorul (sau perechea de senzori!) potrivit, împreună cu argumentul: „ușa unui magazin trebuie să se deschidă singură” · „drona trebuie să stea nemișcată în aer” · „aspiratorul nu are voie să cadă pe scări” · „sera trebuie udată doar când pământul este uscat ȘI este zi” · „telefonul trebuie să se deblocheze doar la fața ta”. Clasa punctează argumentul, nu doar răspunsul; la misiunile cu ȘI, echipa scrie condiția pe tablă în forma din lecție. Ultimul cartonaș, pentru toți: „robotul vostru din Modulul 5: ce senzori va purta?” Primele idei se notează. Chiar le veți folosi.

Varianta Kahoot: misiunile, sub formă de întrebări cu variante de senzori; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 3.6-J: cinci misiuni; senzorul ales și condiția scrisă în forma „dacă ... și ... atunci ...”
Varianta fără ecrane: chiar jocul principal; cartonașe tipărite din folderul lecției sau misiuni strigate

Provocare: a treia condiție

Becul de curte econom: aprinde-l doar dacă este întuneric ȘI este mișcare ȘI temperatura este peste −10 °C (sub ger năprasnic, becul clasic oricum îngheață; glumim: vrem doar trei senzori într-o condiție). Conectează și senzorul TMP36 din 3.4 și scrie condiția triplă. Da, ȘI se poate înlănțui oricât.

3 întrebări rapide

1. Prin ce seamănă LiDAR-ul cu senzorul ultrasonic pe care l-ai folosit și prin ce diferă?
2. Ce este fuziunea senzorilor și de ce nu se bazează roboții pe un singur senzor?
3. Când e adevărată o condiție legată cu operatorul ȘI?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

LiDAR: măsurare cu ecou de lumină laser; sute de mii de distanțe pe secundă = o hartă

cameră de adâncime: cameră care vede și distanța fiecărui punct din imagine

accelerometru / giroscop: senzorii mișcării proprii: înclinare și accelerații / rotire

encoder: numărătorul de rotații al unui ax; kilometrajul roboților

fuziunea senzorilor: combinarea mai multor senzori într-o singură concluzie

SLAM: cartografiere și localizare simultană: harta se construiește din mers

operatorul ȘI (and / &&): leagă condiții; adevărat doar când toate sunt adevărate

Ce trebuie să știi după această lecție

- ✓ Familiile de senzori: proximitate, vedere, forță/atingere, mișcare, poziție și navigație
- ✓ LiDAR = principiul ecoului pe care l-ai exersat, cu lumină în loc de sunet și cu viteză de hartă
- ✓ Roboții serioși nu cred un singur senzor: fuziunea combină mai multe într-o concluzie
- ✓ Operatorul ȘI leagă condiții; becul tău de curte este prima ta fuziune de senzori

Ai terminat Modulul 3!

Robotul tău are acum tot ce îi trebuie: actuatori care mișcă (Modulul 2), senzori care simt și instrucțiunea care decide (Modulul 3). În Modulul 4 le legi pe toate: primele tale dispozitive complete (bariera care se ridică singură când te apropii, măsurată cu ultrasonicul, și ventilatorul cu termostat) și, în a doua jumătate, o lume nouă: proiectarea 3D și imprimantele care transformă desenele în obiecte. Ne vedem acolo!

Din buton la fuziunea senzorilor

3.1 Intrare digitală

buton → 0/1

3.2 Analogic

tensiune → 0–1023

3.3 Prag

lumină → decizie

3.4 Calibrare

număr → grade

3.5 Distanță / mișcare

ecou + PIR

3.6 Fuziune

doi senzori → o decizie

MODULUL 3 · ANEXA ELEVULUI

Răspunsuri la „3 întrebări rapide”

Lecția 3.1

1. Ieșirea scoate semnale la comanda programului; intrarea citește ce tensiune este prezentă; aceiași pini, sens invers.

2. Condiția (adevărată sau falsă), ramura „atunci” și ramura „altfel”; se execută exact una dintre ramuri.

3. Ține pinul-intrare ferm pe 0 cât timp butonul e liber; fără el, pinul „plutește” și citește la întâmplare.

Lecția 3.2

1. Digitalul are trepte fixe (0/1); analogicul variază lin, cu orice valoare din interval.

2. Transformă tensiunea de pe pin într-un număr, proporțional; jumătate din maxim ≈ 512.

3. Ambele citesc pe 0–1023. Scara 0–255 din Modulul 2 era la scriere (PWM pe Arduino), nu la citire.

Lecția 3.3

1. Întuneric = rezistență mare, lumină = rezistență mică; pinul citește tensiuni, nu rezistențe; de aceea îi trebuie împărțitorul cu rezistorul de 10 kΩ.

2. Pragul e granița dintre situații; se alege măsurând valorile reale în ambele situații și luând un număr între ele.

3. Scade: la întuneric fotorezistorul are rezistență mare, punctul de mijloc primește puțină tensiune, iar numărul este mic.

MODULUL 3 · ANEXA ELEVULUI

Răspunsuri la „3 întrebări rapide”

Lecția 3.4

1. Pe Arduino: număr → volți (citire × 5 ÷ 1023), apoi volți → grade ((volți − 0,5) × 100). Pe micro:bit, aceleași două trepte, dar prin calibrare: scazi citirea de la 0 °C și împarți la creșterea pe 100 de grade.

2. O zonă de memorie cu nume pentru o valoare; numele bine alese (citire, volți, grade) fac programul lizibil ca o rețetă.

3. Variația măruntă, întâmplătoare, a citirilor; media mai multor citiri anulează vârfurile între ele.

Lecția 3.5

1. Pentru că timpul măsurat e dus-întors, iar distanța căutată e doar dusul.

2. Ultrasonicul răspunde cu un număr (distanța în centimetri) obținut din durata unui impuls, exact cum servomotorul din 2.6 primea unghiul tot ca durată de impuls; PIR-ul răspunde cu 1/0, ca un buton.

3. Viteza sunetului crește ușor cu temperatura, deci formula distanței se schimbă ușor cu temperatura de afară.

Lecția 3.6

1. Amândoi măsoară cu ecou: trimit un impuls și cronometrează întoarcerea. LiDAR folosește lumină laser, măsoară enorm mai repede și mai precis și, rotindu-se, construiește o hartă.

2. Combinarea mai multor senzori într-o singură concluzie; un singur senzor greșește ușor (camera în întuneric, GPS-ul în tunel).

3. Doar când toate condițiile legate sunt adevărate simultan.

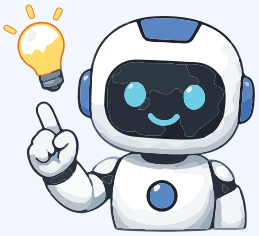
Modulul 4

PRIMUL ROBOT COMPLET

și proiectare 3D

Modulul 4. Primul robot complet și proiectarea 3D

Ai toate componentele: actuatori care mișcă (Modulul 2), senzori care simt și instrucțiuni care decid (Modulul 3). Modulul 4 le unește. În primele trei lecții construiești dispozitive complete (bariera care se ridică singură, ventilatorul cu termostat, lampa inteligentă). Ele sunt cuprinse în volumul de față. În ultimele trei, schimbi instrumentele: înveți cum funcționează imprimantele 3D, îți proiectezi primul obiect în Tinkercad 3D și afli drumul de la model la obiectul tipărit.



Cum e împărțit modulul

Lecțiile 4.1–4.3 · Integrare: lanțul complet simte → decide → acționează, pe cele două trasee obișnuite (Micro:bit / Arduino). Nicio componentă nouă; doar tot ce știi, pus la lucru împreună.

Lecțiile 4.4–4.6 · Proiectare 3D: în partea a doua a modulului, fără trasee și fără plăci: este partea comună tuturor. Acolo rămâi în același cont Tinkercad, dar treci din secțiunea Circuits în secțiunea 3D Designs. La final, obiectele proiectate de tine pot deveni chiar piese pentru robotul echipei tale din Modulul 5.

LEGENDA CULORILOR · ACELEAȘI ÎN TOT MANUALUL

Teorie · Practică Tinkercad · Atenție / Depanare · Provocare / Joc

LECȚIA 4.1 **Bariera automată de parcare: primul robot complet**

Schema-bloc și pseudocodul · senzor + decizie + servomotor, cap-coadă

LECȚIA 4.2 **Ventilatorul cu termostat**

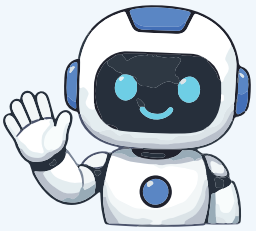
„Altfel dacă”: mai multe ramuri · temperatura comandă viteza motorului

LECȚIA 4.3 **Lampa inteligentă și comanda manuală**

Operatorul SAU · trei intrări, o lampă · turul automatizărilor

LECȚIA 4.1 · 50 DE MINUTE

Bariera automată de parcare: primul robot complet



Bariera ta a funcționat pe bază de timp în 2.6: sus, jos, sus, jos, după un tipar fix. În 3.1 ai învățat cum ascultă o placă de o intrare. Astăzi cele două se întâlnesc: bariera se ridică singură când sesizează mașina. Simte, decide, acționează: lanțul complet, construit de tine.

50 MIN · DIFICULTATE ●●● · INTEGRARE COMPLETĂ · AMBELE TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să desenezi schema-bloc a unei automatizări: intrare → decizie → ieșire
- Să scrii pseudocodul înainte de cod: planul în limba română
- Să afli pe planșă ce unghi ridică bariera, înainte de a scrie decizia
- Să construiești bariera care se ridică singură la apropierea „mașinii”

SECȚIUNEA 01 · TEORIE

Întâi planul, apoi firele

DEJA CUNOȘTI (DIN MODULELE 2 ȘI 3)

Servomotorul primește un unghi și îl menține (2.6), iar unghiul i se transmite prin lățimea impulsului; tabelul de atunci: 0° ≈ 26 · 90° ≈ 77 · 180° ≈ 128. Senzorul ultrasonic raportează distanța, PIR raportează mișcarea (3.5). Instrucțiunea „dacă” alege ramura (3.1). Astăzi nu apare nicio componentă nouă; apare metoda.

Când componentele sunt multe, inginerii nu încep cu firele, ci cu două desene. Primul este schema-bloc: trei căsuțe legate cu săgeți: INTRAREA (ce senzor?), DECIZIA (ce condiție?), IEȘIREA (ce actuator?). Al doilea este pseudocodul: programul scris în limba română, pe puncte, înainte de orice bloc sau rând de Python. Dacă planul este limpede pe hârtie, codul devine o simplă traducere; dacă planul șchiopătează, niciun cod nu îl salvează. Ambele desene vor fi obligatorii în documentația proiectului din Modulul 5; astăzi le exersezi pe un caz cunoscut.

Planul barierei, pe hârtie

Schema-bloc: SENZOR DE PREZENȚĂ → „e o mașină aproape?” → SERVOMOTOR (barieră)

Pseudocod: repetă la nesfârșit: citește senzorul · dacă mașina e aproape, ridică bariera (cornul vertical) · altfel, coboară-o (cornul orizontal, peste drum) · pauză scurtă. Observă că planul nu conține niciun număr: gradele se aleg abia pe planșă, când vezi cum stă cornul.

SECȚIUNEA 02 · PRACTICĂ

Construim bariera, pe traseul tău

Aceeași schemă-bloc, doi senzori de prezență diferiți; fiecare traseu îl folosește pe cel care i se potrivește: la Arduino, senzorul ultrasonic (cu blocul lui dedicat, din 3.5); la Micro:bit, senzorul PIR (citire digitală simplă). Ambele bariere sunt roboți completi.

⚠ Întâi privește cornul, apoi alege unghiurile

Servomotorul nu știe ce este o barieră. El primește un unghi al axului și îl menține; atât. Dacă unghiul acela înseamnă „sus” sau „jos” nu decide codul, ci felul în care stă servomotorul pe planșă.

Pornește simularea și privește cornul micro-servomotorului din Tinkercad la cele trei unghiuri cunoscute din 2.6:
• 0° (valoarea 26) → cornul vertical, ridicat → BARIERA SUS
• 90° (valoarea 77) → cornul culcat, orizontal → BARIERA JOS, peste drum
• 180° (valoarea 128) → cornul iar vertical, dar coborât: a trecut dincolo de orizontală

Surpriza lecției: numărul mai mare NU înseamnă „mai sus”. Pe măsură ce unghiul crește, cornul coboară. De aceea bariera lucrează între 90° și 0°: stă culcată la 90° și se ridică mergând spre 0°, exact ca la intrarea într-o parcare. Cu perechea 90°–180°, cornul ar ajunge tot vertical, dar coborând, ca și cum bariera s-ar prăbuși prin asfalt.

Regula rămâne valabilă pentru orice mecanism pe care îl vei construi: unghiurile se aleg după cum arată piesa pe planșă, nu după cum sună numărul.

TRASEUL A · MICRO:BIT

PIR → P1 · servomotorul → P0 · bariera lucrează între 90° (jos) și 0° (sus)

CIRCUIT DE START

▶ Deschide circuitul de start 4.1-A (Micro:bit)

Placa Micro:bit, senzorul PIR și micro-servomotorul, așezate pe planșă, fără niciun fir și fără nicio linie de cod. Apasă „Copy and Tinker”, apoi cablează tu, după pașii de mai jos.

1 · Cablează cele două lanțuri

Nicio componentă nouă: sunt montajele din 3.5 și 2.6, așezate pe aceeași planșă.

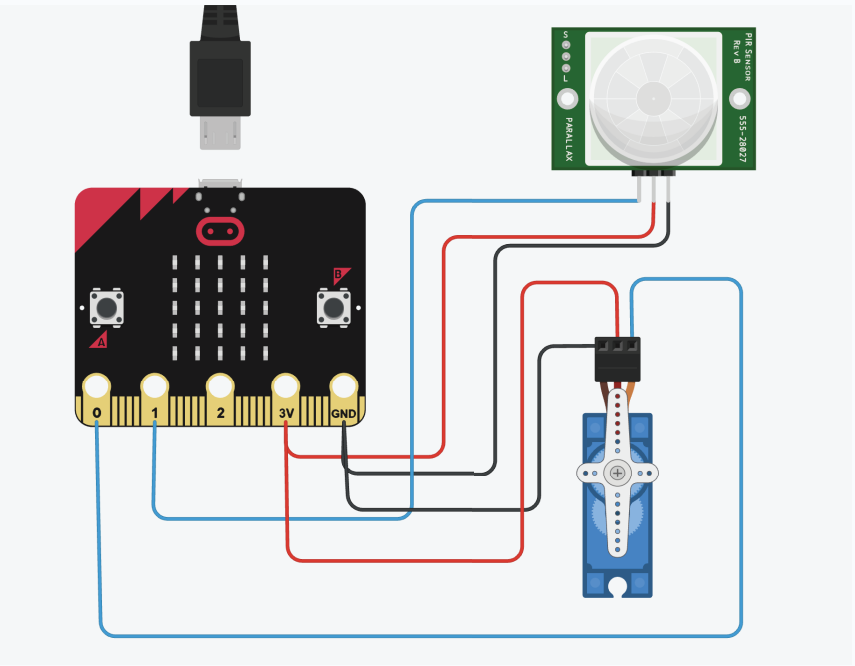
SENZORUL · PIR-UL DIN 3.5

• alimentare (3V) → pinul de alimentare al PIR-ului
• GND → GND-ul PIR-ului
• semnalul PIR-ului → P1

ACTUATORUL · SERVOMOTORUL DIN 2.6

• firul roșu → 3V (de aici își ia forța)
• firul maro sau negru → GND
• firul portocaliu → P0 (pe aici primește comanda; în capturi jumperul este albastru, dar el pleacă din firul portocaliu al servomotorului)

Două lanțuri scurte: unul simte, celălalt acționează. Decizia care le leagă nu se cablează; ea stă în cod. Firul roșu al servomotorului nu se conectează niciodată la un pin de comandă: pinul furnizează semnal, nu curent de lucru.



☐ Ambele lanțuri sunt cablate

2 · Setează „limba” servomotorului și află care unghi ridică bariera

Ca în 2.6, placa Micro:bit nu are bloc de servomotor: îi ceri mai întâi pinului P0 să trimită câte un impuls la fiecare 20 ms, apoi scrii pe scara 0–1023 valorile din tabelul Lecției 2.6: 0° ≈ 26 · 90° ≈ 77 · 180° ≈ 128. Rândul cu perioada se scrie o singură dată, ÎNAINTEA buclei, nu în interiorul ei.

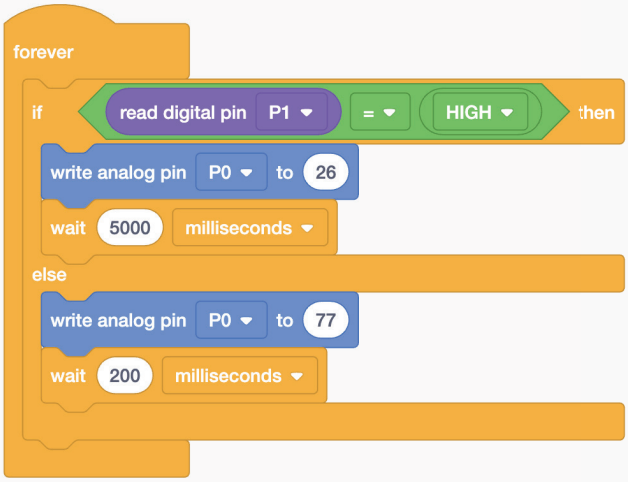
Înainte de decizie, fă un program scurt, doar ca să vezi pozițiile: forever → write analog pin P0 to 77 → pause 2000 → write analog pin P0 to 26 → pause 2000. Pornește simularea și privește nu doar capetele, ci și drumul dintre ele: la 77 cornul stă culcat (bariera jos), la 26 se ridică vertical (bariera sus), iar între cele două valori trebuie să-l vezi urcând, nu coborând.

Dacă la tine rezultatul este diferit (deoarece ai rotit servomotorul pe planșă), notează propriile două valori și folosește-le mai departe. Planșa ta comandă, nu manualul.

☐ Știu ce valoare înseamnă „sus” și ce valoare înseamnă „jos” pe planșa mea

3 · Scrie decizia

Acum pseudocodul se traduce singur. În blocuri: forever → if (read digital pin P1 = HIGH) → write analog pin P0 to 26 → pause 5000 → else → write analog pin P0 to 77 → pause 200. Apoi comută pe Python.
Pornește simularea și execută clic lângă lentila PIR-ului: apare punctul de „mișcare” pe care îl plimbi prin fața senzorului. Bariera trebuie să urce la mișcare, să stea sus 5 secunde și să coboare singură după aceea. Un amănunt: în blocuri, pauza de 200 ms stă în ramura else, iar în Python am scos-o în afară, la sfârșitul buclei. Bariera se comportă la fel, dar în afară este locul ei firesc, fiindcă privește întreaga buclă.



☐ Bariera urcă la mișcare și coboară singură după 5 secunde

BARIERA AUTOMATĂ · TRADUCEREA PSEUDOCODULUI **PYTHON · MICRO:BIT**

```
# Lecția 4.1: bariera automată de parcare
# (varianta Tinkercad / MakeCode)

# „Limba” servomotorului, din Lecția 2.6: un impuls la fiecare 20 ms.
# Se scrie o singură dată, înaintea buclei.
pins.analog_set_period(AnalogPin.P0, 20000)   # 20000 microsecunde = 20 ms

# Cele două poziții ale barierei, verificate de tine la pasul 2.
# Numele spun intenția, numerele spun geometria planșei.
SUS = 26   # 0° → cornul vertical → bariera ridicată
JOS = 77   # 90° → cornul culcat → bariera coborâtă peste drum

def on_forever():
    # SIMTE: PIR-ul se citește exact ca un buton: 1 sau 0
    if pins.digital_read_pin(DigitalPin.P1) == 1:
        pins.analog_write_pin(AnalogPin.P0, SUS)   # ACȚIONEAZĂ: bariera sus
        basic.pause(5000)   # stă sus 5 secunde; mașina trece
    else:
        pins.analog_write_pin(AnalogPin.P0, JOS)   # altfel: bariera jos
    basic.pause(200)

basic.forever(on_forever)   # forever din blocuri = acest rând în Python
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Ultrasonicul → pinul 13 (ca în 3.5) · servomotorul → pinul 9 · aceleași poziții: 90° jos, 0° sus

CIRCUIT DE START

▶ Deschide circuitul de start 4.1-B (Arduino)

Placa Arduino Uno, senzorul ultrasonic și micro-servomotorul, așezate pe planșă, fără fire și fără cod. Apasă „Copy and Tinker".

1 · Cablează senzorul și actuatorul

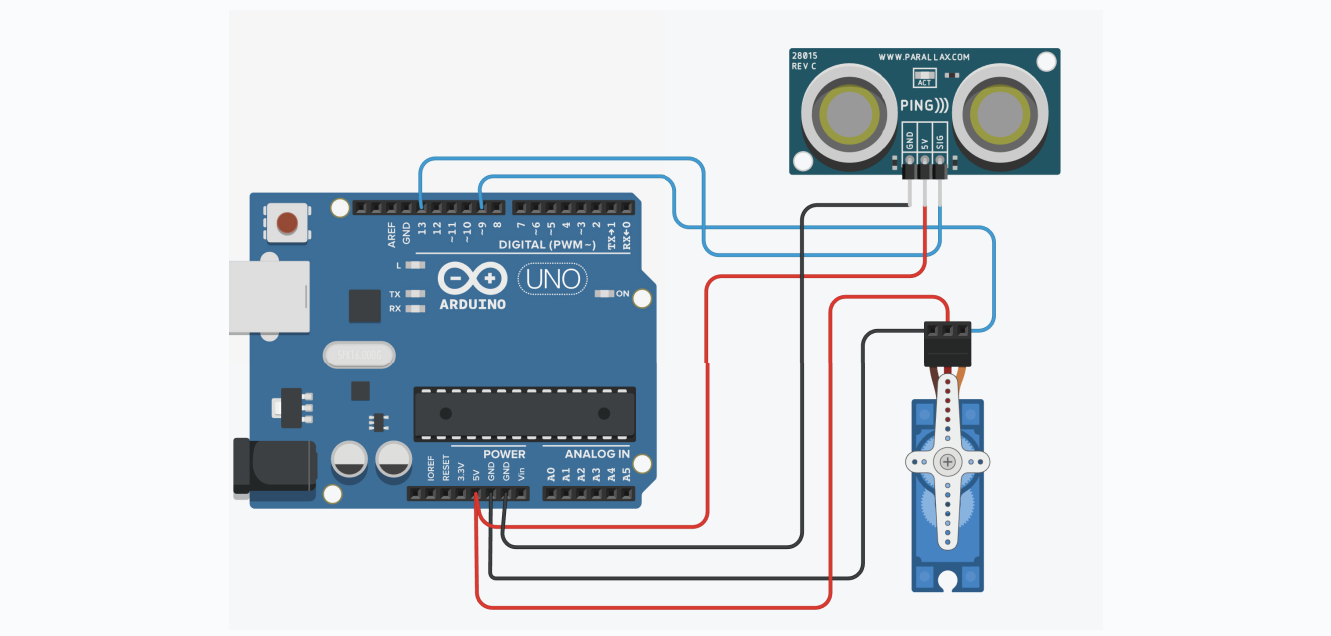
SENZORUL · ULTRASONICUL DIN 3.5

- alimentare (5V) → VCC-ul senzorului
- GND → GND-ul senzorului
- SIG (semnalul) → pinul 13

Tinkercad așază senzorul implicit în varianta cu trei terminale, cu un singur fir de semnal: exact cea din 3.5. Dacă a ta are patru terminale (trigger și echo separate), schimb-o din caseta de proprietăți.

ACTUATORUL · SERVOMOTORUL DIN 2.6

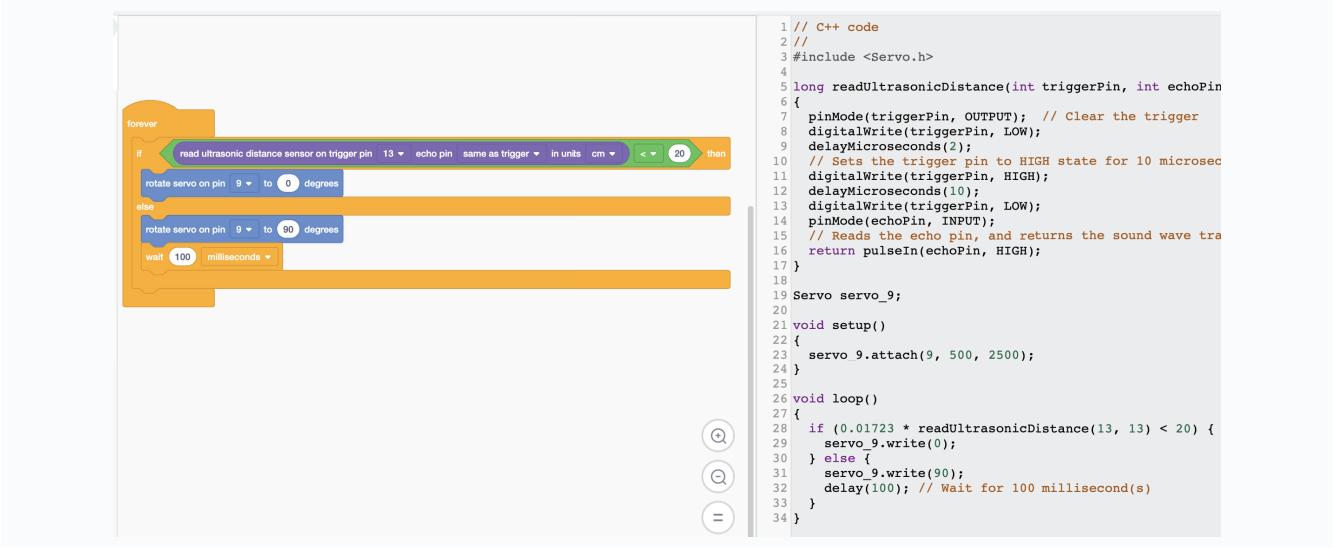
- firul roșu → 5V
- firul maro sau negru → GND
- firul portocaliu → pinul 9



☐ Ambele lanțuri sunt cablate

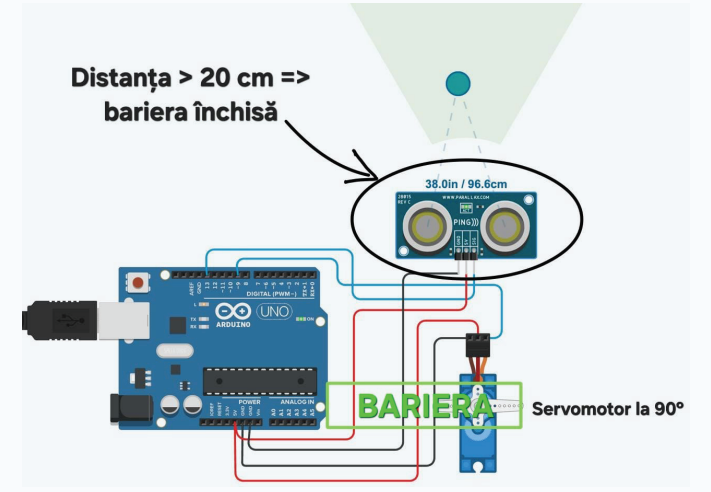
2 · Tradu pseudocodul în blocuri și testează

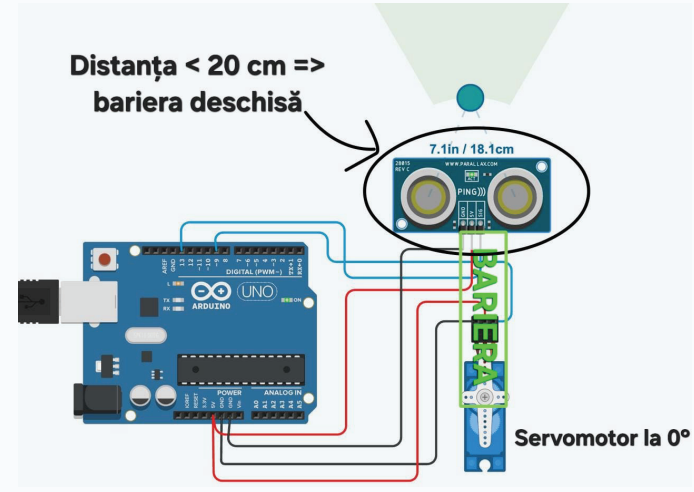
Apasă „Code” și alege modul „Blocks + Text”. Aici blocul de servomotor cere unghiul direct în grade, deci scrii chiar 90 și 0, fără traducerea în valori de impuls.
Verifică mai întâi pozițiile, exact ca pe celălalt traseu: forever → rotate servo on pin 9 to 90 degrees → wait 2 secs → rotate servo on pin 9 to 0 degrees → wait 2 secs. La 90° cornul stă culcat (bariera jos), la 0° este ridicat (bariera sus), iar între ele îl vezi urcând, nu coborând.
Apoi bucla adevărată, în ordinea pseudocodului: forever → if (read ultrasonic distance sensor on trigger pin 13, echo pin same as trigger, in units cm) < 20 → rotate servo on pin 9 to 0 degrees → else → rotate servo on pin 9 to 90 degrees → wait 100 ms. În captură, pauza este așezată în interiorul ramurii else, și tot acolo o vezi în vitrina C++, ca delay(100). În listingul comentat de mai jos am scos-o în afară, după perechea if / else: este locul ei firesc, fiindcă pauza privește întreaga buclă, nu o singură ramură. Bariera se comportă la fel în ambele variante. Cele două blocuri fac singure munca grea: unul măsoară timpul ecoului și îl dă în centimetri (ca în 3.5), celălalt duce cornul la unghiul cerut (ca în 2.6).
Pornește simularea și trage obiectul-țintă al senzorului spre el: sub 20 cm bariera urcă, îndepărtezi ținta și bariera coboară. Trage ținta, nu senzorul. În vitrina C++ din dreapta recunoști tot: Servo.h, funcția ecoului și perechea if / else.




```
// C++ code
//
#include <Servo.h>

long readUltrasonicDistance(int triggerPin, int echoPin
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microsec
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  // Reads the echo pin, and returns the sound wave tra
  return pulseIn(echoPin, HIGH);
}

Servo servo_9;

void setup()
{
  servo_9.attach(9, 500, 2500);
}

void loop()
{
  if (0.01723 * readUltrasonicDistance(13, 13) < 20) {
    servo_9.write(0);
  } else {
    servo_9.write(90);
    delay(100); // Wait for 100 millisecond(s)
  }
}
```

Demonstrație pe traseul B; cele două poziții ale barierei sunt aceleași și pe micro:bit:





☐ Bariera urcă singură la apropierea țintei

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ C++ · ARDUINO

```
// C++ code
// Lecția 4.1: bariera automată de parcare (varianta Arduino)

#include <Servo.h>   // biblioteca servomotorului, din 2.6

Servo bariera;   // numele pe care i-l dăm servomotorului
int distanta = 0;   // distanța citită, în centimetri

// Cele două poziții, verificate pe planșă la pasul 2
const int SUS = 0;   // cornul vertical → bariera ridicată
const int JOS = 90;   // cornul culcat → bariera coborâtă peste drum

// Funcția care pune senzorul să strige și să asculte ecoul (din 3.5)
long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerPin, HIGH);   // „strigătul”: impuls de 10 microsecunde
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  return pulseIn(echoPin, HIGH);   // cât a durat până s-a întors ecoul
}

void setup() {
  bariera.attach(9, 500, 2500);   // servomotorul ascultă de pinul 9
}

void loop() {
  // 0.01723 transformă timpul dus-întors direct în centimetri
  distanta = 0.01723 * readUltrasonicDistance(13, 13);
  if (distanta < 20) {   // SIMTE și DECIDE: e mașina aproape?
    bariera.write(SUS);   // ACȚIONEAZĂ: bariera se ridică
  } else {
    bariera.write(JOS);   // altfel: bariera stă jos
  }
  delay(100);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT

PIR-ul răspunde doar „este mișcare / nu este”: bariera ta reacționează la orice trecere, dar nu cunoaște distanța.

ARDUINO UNO

Ultrasonicul măsoară distanța exactă: bariera urcă doar când mașina este chiar în dreptul ei, ca la mall.

Concluzia: aceeași schemă-bloc, senzori diferiți, comportamente diferite. Alegerea senzorului ESTE proiectarea: prima decizie din orice fișă de proiect.

SECȚIUNEA 03 · DEPANARE

⚠ Bariera nu ascultă?

Regula de aur a integrării: testează fiecare jumătate separat înainte de a le judeca împreună. 1) Rulează mai întâi doar programul senzorului (afișarea din 3.2 / 3.5): citește? 2) Apoi doar servomotorul, cu ciclul din 2.6: se mișcă? 3) Abia apoi unește-le. 4) Bariera se comportă invers sau se ridică prin partea greșită? Nu este o greșeală de logică, ci de unghiuri: în Tinkercad, unghiul mai mare COBOARĂ cornul. 0° = ridicat, 90° = culcat, 180° = coborât dincolo de orizontală. Rulează mai întâi programul de verificare de la pasul 2 și abia apoi scrie decizia. 5) La micro:bit: rândul pins.analog_set_period(AnalogPin.P0, 20000) este scris ÎNAINTEA funcției on_forever, nu în interiorul ei? 6) Cornul se mișcă, dar nu ajunge la capăt → ajustează valorile 26 și 77 cu ±5, ca în 2.6. 7) Bariera „tremură” jos-sus → mărește pauza finală.

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, pe echipe de câte 3)

Trei elevi = cele trei căsuțe din schema-bloc: Senzorul, Decizia, Actuatorul. Profesorul strigă o automatizare din viața reală („ușa de la mall”, „lumina de pe casa scării”, „uscătorul de mâini”, „semaforul cu buton de pieton”), iar echipa o joacă în 30 de secunde: Senzorul anunță ce simte („văd o persoană la 1 metru!”), Decizia strigă condiția („distanța sub 2 metri: adevărat!”), Actuatorul execută mișcarea. Clasa validează: lanțul este corect și complet? Echipele care inventează automatizări noi pentru colegi primesc punct dublu.

Varianta Kahoot: automatizări de recunoscut: „ce este senzorul, ce este decizia, ce este actuatorul?”; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 4.1-J: patru automatizări; completează schema-bloc și pseudocodul fiecăreia
Varianta fără ecrane: chiar jocul principal; necesită trei scaune în fața clasei și imaginație

Provocare: bariera de serie mare

Nivel 1 · Semnalizare. Adaugă LED-ul roșu care clipește cât timp bariera este jos și LED-ul verde aprins cât timp este sus (cu rezistorii lor de protecție, pe care îi cunoști).
Nivel 2 · Sunet. Piezo-ul din 3.5: un „bip” scurt la fiecare ridicare.
Nivel 3 · Coborâre politicoasă. Bariera coboară în trepte de 10°, de la 0° spre 90°, nu dintr-o singură mișcare; nicio barieră adevărată nu trântește.

3 întrebări rapide

1. Care sunt cele trei căsuțe ale schemei-bloc și ce întrebare răspunde fiecare?
2. La ce folosește pseudocodul, dacă tot scriem codul după el?
3. Prin ce diferă bariera cu PIR de bariera cu ultrasonic?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

schemă-bloc: desenul automatizării: intrare → decizie → ieșire, cu săgeți

pseudocod: programul scris în limba română, pe puncte, înainte de cod

integrare: unirea senzorilor, deciziilor și actuatorilor într-un dispozitiv complet

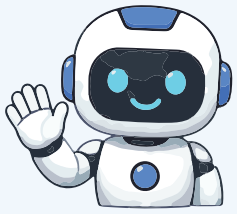
automatizare: dispozitiv care își îndeplinește sarcina singur, după reguli scrise de om

Ce trebuie să știi după această lecție

- ✓ Întâi schema-bloc și pseudocodul, apoi firele și codul: ordinea inginerului
- ✓ Robotul complet = senzor + decizie + actuator, în aceeași buclă
- ✓ Unghiul e o poziție a axului, nu o intenție: în Tinkercad unghiul mai mare coboară cornul, deci verifică pozițiile înainte de a scrie decizia
- ✓ Alegerea senzorului schimbă comportamentul: PIR simte trecerea, ultrasonicul știe distanța
- ✓ La depanare, fiecare jumătate se testează separat înainte de a fi judecate împreună

LECȚIA 4.2 · 50 DE MINUTE

Ventilatorul cu termostat



Laptopul tău o face zilnic: când se încălzește, pornește ventilatorul; când fierbe, îl turează. În această lecție construiești exact acel mecanism și înveți instrucțiunea cu mai multe ramuri: „altfel dacă”.

50 MIN · DIFICULTATE ●●● · TEMPERATURĂ + MOTOR · AMBELE TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să scrii o decizie cu trei ramuri, folosind „altfel dacă” (elif)
- Să legi termometrul din 3.4 de motorul cu tranzistor din 2.3
- Să dai motorului alimentare proprie și să explici de ce are nevoie de ea
- Să reglezi treptele de viteză cu praguri de temperatură alese de tine

SECȚIUNEA 01 · TEORIE

„Altfel dacă”: decizia în trepte

DEJA CUNOȘTI (DIN LECȚIILE 2.3, 2.5 ȘI 3.4)

Motorul se comandă prin tranzistor, viteza lui prin PWM, iar TMP36 dă gradele prin calcul: cu formula din fișa tehnică pe Arduino, cu regula de trei din calibrarea ta pe Micro:bit. Toate trei intră azi în același program.

„Dacă… altfel” desparte lumea în două. Dar termostatul are nevoie de trei situații: răcoare (oprit), cald (viteză medie), foarte cald (viteză maximă). Pentru asta, între „dacă” și „altfel” se pot adăuga oricâte trepte intermediare: „altfel dacă” : în Python, cuvântul elif; în blocuri, butonul care adaugă o ramură „else if”. Regula de execuție rămâne una singură: condițiile se verifică de sus în jos, iar programul intră în prima ramură adevărată și le sare pe toate celelalte. De aceea ordinea contează: pragul cel mai mare se scrie primul.

Planul termostatului

Schema-bloc: TMP36 → „în ce treaptă suntem?” → MOTOR (prin tranzistor)

Pseudocod: repetă: citește gradele · dacă grade > 30, motor la maximum · altfel dacă grade > 26, motor la jumătate · altfel, motor oprit · pauză. Pragurile 26 și 30 sunt propunerea noastră; la final le reglezi după preferință, ca la orice termostat.

SECȚIUNEA 02 · PRACTICĂ

Trei trepte, comandate de căldură

Pe planșă se întâlnesc două circuite deja construite de tine: termometrul TMP36 din 3.4, conectat la pinul analogic (P0 / A0), și lanțul pin → rezistor 1 kΩ → tranzistor → motor, cu dioda lui de protecție, din 2.3, conectat la un pin care acceptă PWM (P1 / pinul 9). O singură noutate, dar una importantă: pe traseul Micro:bit, motorul nu se mai alimentează din placă, ci din bateriile lui. De ce: imediat mai jos.

CIRCUIT DE START

▶ Deschide circuitul de start 4.2-A (Micro:bit)

Placa Micro:bit, senzorul TMP36, tranzistorul, rezistorul de 1 kΩ, dioda, motorul și un pachet de baterii (4 × AA), așezate pe planșă, neconectate și fără cod. Apasă „Copy and Tinker”.

CIRCUIT DE START

▶ Deschide circuitul de start 4.2-B (Arduino)

Aceleași componente, cu placa Arduino Uno în locul celei Micro:bit, și fără pachetul de baterii: aici motorul se alimentează din pinul de 5V al plăcii. Neconectate și fără cod. Apasă „Copy and Tinker”.

1 · Cablează cele două jumătăți și testează-le separat

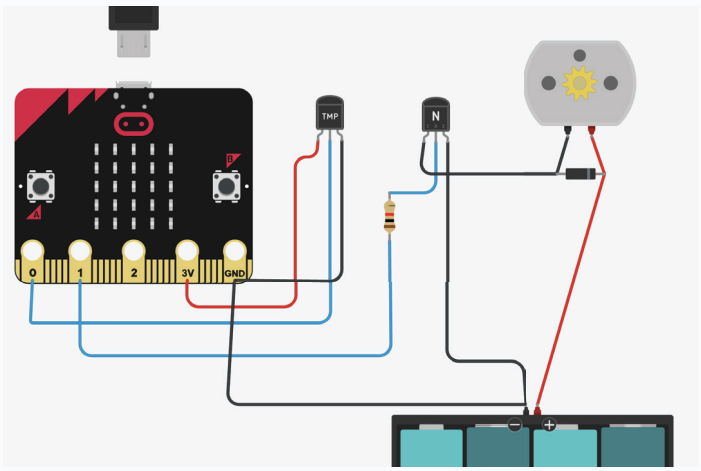
TERMOMETRUL · TMP36 DIN 3.4 (ține senzorul cu fața plată spre tine)
• terminalul din stânga → alimentare (3V / 5V)
• terminalul din mijloc → P0 / A0
• terminalul din dreapta → GND
MOTORUL PE TRASEUL A (Micro:bit) · cu alimentare proprie
• P1 → rezistor 1 kΩ → baza tranzistorului (terminalul din mijloc)
• plusul (+) al bateriilor → o bornă a motorului
• cealaltă bornă a motorului → colectorul tranzistorului
• emitorul tranzistorului → minusul (−) al bateriilor
• minusul (−) al bateriilor → GND-ul plăcii Micro:bit
• dioda, în paralel cu motorul, cu dunga spre plusul bateriilor
MOTORUL PE TRASEUL B (Arduino) · direct din placă
• pinul 9 → rezistor 1 kΩ → baza tranzistorului
• emitorul → GND
• 5V → motor → colectorul
• dioda, în paralel cu motorul, cu dunga spre 5V

Ultimul fir din listă este cel mai important și cel mai ușor de uitat: fără el, cele două circuite nu au nicio referință comună, iar tranzistorul nu are față de ce să compare comanda primită în bază.

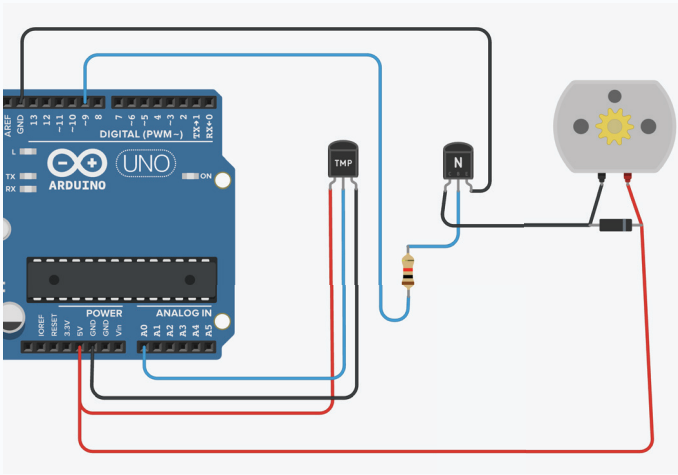
Regula din 4.1: mai întâi termometrul singur (programul din 3.4: apar gradele?), apoi motorul singur (treptele din 2.5: se rotește la jumătate de scară?). Doar după ambele bife treci la programul întreg.

☐ Fiecare jumătate funcționează separat

Micro:bit:



Arduino:



⚠ De ce are motorul baterii proprii pe Micro:bit

Până acum, în 2.3 și 2.5, motorul a funcționat fără probleme din pinul de 3V al plăcii. Nici la bariera din 4.1 nu ne-a deranjat: servomotorul a stat tot pe 3V, iar PIR-ul răspunde doar „da / nu”, deci nu avea ce să falsifice. Astăzi, pentru prima dată, pe aceeași planșă stă un senzor care MĂSOARĂ, și aici lucrurile se schimbă.

Pinul de 3V al plăcii Micro:bit poate furniza în jur de 90 mA. Un motor cere mult mai mult, mai ales în prima clipă, când pornește din repaus. Când motorul absoarbe curent, tensiunea de pe pin scade puțin, iar Micro:bit-ul își raportează citirile analogice tocmai față de acea tensiune de referință. Rezultatul: numărul citit de la TMP36 se schimbă în funcție de cât curent absoarbe motorul.

Iar motorul absoarbe curent deoarece așa a decis programul, pe baza aceleiași citiri. Măsurătoarea ajunge astfel să depindă de propria ei decizie: o buclă de reacție nedorită, în care termostatul pornește brusc, rămâne agățat sus și coboară abia cu multe grade mai jos decât ar trebui.

Soluția este regula din 2.3, dusă până la capăt: motorul își ia forța din alimentarea lui, nu din placă. Bateriile alimentează motorul, Micro:bit-ul comandă doar baza tranzistorului prin rezistorul de 1 kΩ, iar cele două circuite se ating într-un singur punct: masa comună. Pe Arduino problema nu apare: pinul lui de 5V este alimentat prin USB și suportă fără dificultate câteva sute de miliamperi.

TRASEUL A · MICRO:BIT

TMP36 → P0 · motorul → P1, alimentat din baterii · gradele, prin calibrarea făcută de tine în 3.4

A2 · Recuperează-ți calibrarea din 3.4

Aici este capcana lecției. Modelul de Micro:bit din Tinkercad nu respectă fișa tehnică a senzorului TMP36; ai descoperit acest lucru chiar tu, în 3.4. De aceea gradele nu se calculează cu formula voltților, ci cu regula de trei din cele două numere măsurate de tine: citirea cu glisorul senzorului pe 0 °C și citirea cu glisorul pe 100 °C.
La noi au rezultat 223 și 480, deci grade = (citire − 223) × 100 ÷ 257. Deschide caietul de la 3.4 și folosește propriile numere, dacă au rezultat altele.

☐ Am cele două numere de calibrare în față

A3 · Construiește decizia în trepte și testeaz-o

Ordinea este cea din pseudocod, de sus în jos: mai întâi pragul mare (grade > 30 → scrie 1023 pe P1), apoi pragul mic (grade > 26 → 512), iar în „altfel” oprești motorul (0). Gradele le calculezi cu regula de trei de la pasul A2, cu numerele tale de calibrare.
Când scrii codul, compară-l cu programul de mai jos, rând cu rând: vei găsi elif exact acolo unde ai apăsat „+”. Pornește simularea, execută clic pe TMP36 și plimbă-i glisorul de temperatură: sub 26 °C motorul stă, între 26 și 30 se rotește la jumătate de scară (512 din 1023), peste 30 turează la maximum.

☐ Cele trei trepte răspund la glisor

TERMOSTATUL · PRIMA DECIZIE ÎN TREPTE — **PYTHON · MICRO:BIT**

```
# Lecția 4.2: ventilatorul cu termostat
# (varianta Tinkercad / MakeCode)

# Cele două numere măsurate de tine în Lecția 3.4.
CITIRE_LA_0 = 223   # numărul citit cu glisorul pe 0 °C
CITIRE_LA_100 = 480   # numărul citit cu glisorul pe 100 °C

# Impulsuri dese pentru motor: 1 ms în loc de cele 20 ms implicite.
# Fără rândul acesta, motorul se smucește la treapta de mijloc.
pins.analog_set_period(AnalogPin.P1, 1000)   # 1000 microsecunde = 1 ms

PRAG_MARE = 30   # peste atâtea grade: viteză maximă
PRAG_MIC = 26   # peste atâtea grade: jumătate de viteză

def on_forever():
    citire = pins.analog_read_pin(AnalogPin.P0)   # treapta 0: numărul brut
    # regula de trei din 3.4, care transformă numărul în grade
    grade = (citire - CITIRE_LA_0) * 100 / (CITIRE_LA_100 - CITIRE_LA_0)
    if grade > PRAG_MARE:   # pragul mare se verifică PRIMUL
        pins.analog_write_pin(AnalogPin.P1, 1023)   # foarte cald: viteză maximă
    elif grade > PRAG_MIC:   # „altfel dacă”: treapta de mijloc
        pins.analog_write_pin(AnalogPin.P1, 512)   # cald: jumătate din scară
    else:
        pins.analog_write_pin(AnalogPin.P1, 0)   # răcoare: oprit
    basic.pause(500)

basic.forever(on_forever)
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

TMP36 → A0 · motorul → pinul 9 · gradele, direct din fișa tehnică

B2 · Trage glisorul de temperatură și privește treptele

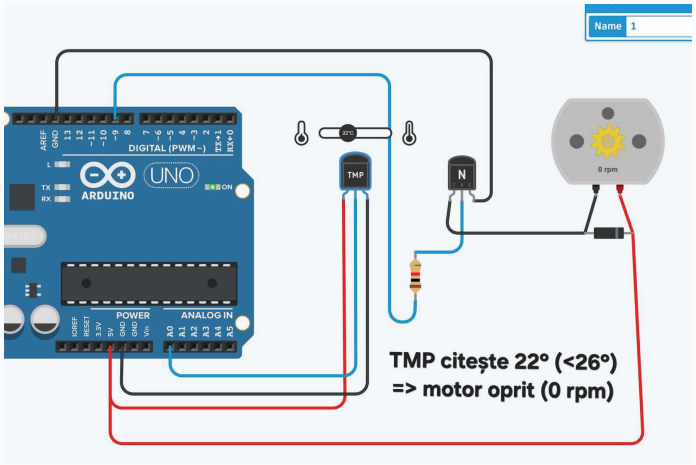
Pe Arduino, formula din fișa tehnică funcționează; ai verificat-o chiar tu, în provocarea Lecției 3.4: volți = citire × 5 ÷ 1023, apoi grade = (volți − 0,5) × 100.

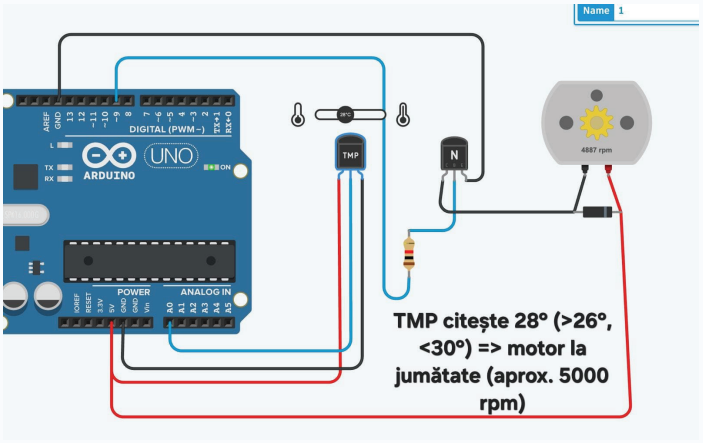
Aici însă nu mai construim blocuri, ci scriem direct în panoul de text: blocurile Arduino din Tinkercad lucrează numai cu numere întregi, iar formula de mai sus are nevoie de zecimale; este aceeași limită pe care ai întâlnit-o în 3.4. Apasă „Code", alege modul „Blocks + Text" și scrie programul din caseta următoare în partea dreaptă.

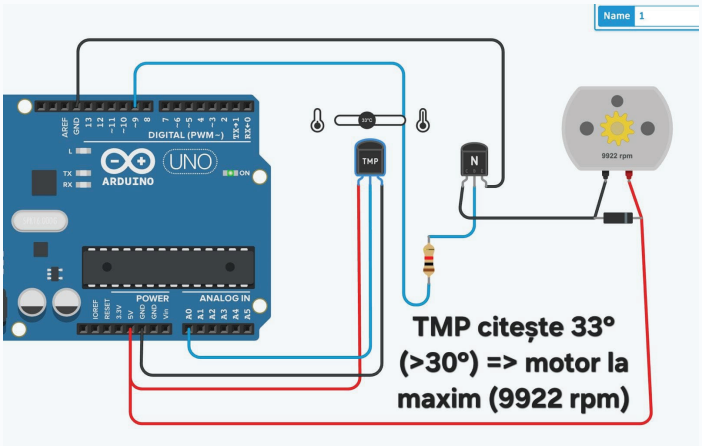
Atenție la scară: pe Arduino, PWM-ul funcționează pe scara 0-255, nu până la 1023 (Lecția 2.5). Jumătate de viteză înseamnă 128, nu 512.

Pornește simularea, execută clic pe TMP36 și plimbă-i glisorul de temperatură: sub 26 °C motorul stă, între 26 și 30 funcționează la jumătate (verifică turația afișată!), peste 30 turează la maximum.

Demonstrație pe traseul B; pe Micro:bit treptele arată la fel; se schimbă doar numerele scării (0 · 512 · 1023 în loc de 0 · 128 · 255):







☐ Treptele răspund la glisor

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ — **C++ · ARDUINO**

```
// C++ code
// Lecția 4.2: ventilatorul cu termostat (varianta Arduino)

int citire = 0;
float volti = 0;   // float = număr cu virgulă, ca în 3.4
float grade = 0;

void setup() {
  Serial.begin(9600);   // jurnalul, ca să vezi gradele
  pinMode(A0, INPUT);
  pinMode(9, OUTPUT);   // pinul 9 are ~, deci poate genera PWM (Lecția 2.5)
}

void loop() {
  citire = analogRead(A0);   // treapta 0: numărul brut
  volti = citire * 5.0 / 1023;   // treapta 1: numărul → volți
  grade = (volti - 0.5) * 100;   // treapta 2: voltii → grade
  Serial.println(grade);

  if (grade > 30) {   // pragul mare se verifică PRIMUL
    analogWrite(9, 255);   // foarte cald: viteză maximă
  } else if (grade > 26) {   // „altfel dacă” se scrie în două cuvinte
    analogWrite(9, 128);   // cald: jumătate din scara 0–255
  } else {
    analogWrite(9, 0);   // răcoare: oprit
  }
  delay(500);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT
Blocurile arată treptele îmbrăcate una în alta, iar în Python cele trei ramuri elif se citesc pe verticală, ca o listă; gradele cer însă mai întâi calibrarea ta din 3.4, iar motorul cere baterii separate.

ARDUINO UNO
Aici blocurile nu ajung: cele din Tinkercad lucrează numai cu numere întregi, iar formula termometrului cere zecimale; de aceea scrii direct în panoul de text, cu else if. În schimb, gradele ies direct din fișa tehnică, fără calibrare, iar pinul de 5V alimentează singur motorul.

Concluzia: deciziile în trepte sunt teritoriul unde codul-text începe să câștige lizibil. Cu cât automatizarea crește, cu atât Python-ul respiră mai ușor.

SECȚIUNEA 03 · DEPANARE

⚠ Treptele nu se poartă cum trebuie?

1) Motorul funcționează mereu la maximum → pragurile sunt în ordine inversă: pragul mare trebuie verificat primul (prima ramură adevărată câștigă!). 2) Nu pornește la treapta mică → amintește-ți din 2.5: sub aproximativ 15 % motorul nu învinge inerția; treapta de mijloc trebuie să fie cel puțin 40-50 % din scara plăcii tale: 512 din 1023 pe micro:bit, 128 din 255 pe Arduino. 3) Gradele îngheață → semnalul TMP36 nu este în pinul analogic. 4) Pe micro:bit gradele ies uriașe sau negative → nu ai înlocuit numerele de calibrare din 3.4 cu ale tale.

5) Pe micro:bit, gradele afișate se schimbă singure când pornește motorul, deși nu ai atins glisorul → motorul este încă alimentat din placă. Dă-i bateriile lui, ca la pasul 1. 6) Motorul stă, iar citirile devin haotice → același fenomen, în forma lui severă: motorul blocat absoarbe curentul maxim. 7) Ai baterii, dar motorul nu se mișcă deloc → verifică firul dintre minusul bateriilor și GND-ul plăcii; fără masa comună, comanda din bază nu are nicio referință. 8) Motorul se smucește la treapta de mijloc → lipsește rândul cu analog_set_period(AnalogPin.P1, 1000).

9) Motorul răspunde cu o mică întârziere → este normal și nu se repară: bucla verifică temperatura de două ori pe secundă (basic.pause(500)), iar un motor cu inerție are nevoie de o clipă pentru a urca sau a coborî turația. Dacă vrei un răspuns mai prompt, scurtează pauza la 200.

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (6 minute, toată clasa)

Pe tablă, pragurile lecției: sub 26 → liniște · 26-30 → aplauze rare · peste 30 → aplauze rapide. Profesorul strigă temperaturi („24!”, „28!”, „31!”, „30 fix!”), iar clasa este ventilatorul: răspunde instant cu treapta corectă. Capcanele fac jocul: „26 fix” (nu este PESTE 26: liniște!) și „30 fix” (nu este peste 30: aplauze rare!). Cine greșește treapta explică ce ramură a ales și de ce. Ultimele runde: un elev preia rolul profesorului, iar pragurile se schimbă din mers.

Varianta Kahoot: temperaturi de încadrat în trepte, cu capcane la limite; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 4.2-J: completează tabelul temperatură → ramura aleasă → viteza; plus două praguri de propus pentru camera ta
Varianta fără ecrane: chiar jocul principal; necesită doar o tablă și palme

Provocare: termostatul de casă

Nivel 1 · A patra treaptă. Adaugă „foarte foarte cald” (peste 35°): motor la maximum ȘI un LED roșu de avertizare. Câte ramuri are acum decizia?

Nivel 2 · Iarna. Întoarce logica: sub 18°, aprinde LED-ul „pornește căldura”. Un termostat de centrală e fix ventilatorul tău, cu comparațiile întoarse.

3 întrebări rapide

1. Ce face „altfel dacă” (elif) și în ce ordine se verifică ramurile?
2. De ce se scrie pragul cel mai mare primul?
3. De ce nu punem treapta de mijloc la 20 % din viteză?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

altfel dacă (elif / else if): ramură intermediară între „dacă” și „altfel”

prima ramură adevărată: regula de execuție: programul intră în ea și le sare pe restul

termostat: automatizarea care compară temperatura cu praguri și comandă în consecință

treaptă: o pereche prag → acțiune din decizia în trepte

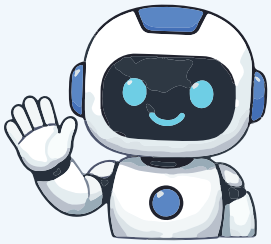
Ce trebuie să știi după această lecție

- ✓ elif adaugă oricâte trepte între „dacă” și „altfel”
- ✓ Condițiile se verifică de sus în jos; câștigă prima adevărată; deci pragul mare stă primul
- ✓ Termostat = formula din 3.4 + treptele PWM din 2.5, într-o singură buclă
- ✓ Un actuator puternic primește alimentare proprie, iar cu placa împarte doar masa; altfel consumul lui falsifică măsurătoarea care îl comandă
- ✓ Pragurile sunt reglaje de proiectare: le alegi, le testezi, le ajustezi

LECȚIA 4.3 · 50 DE MINUTE

Lampa inteligentă și comanda manuală

Becul de curte din 3.6 avea o slăbiciune: nu îl puteai aprinde tu, când doreai. În această lecție îi adaugi comanda manuală cu operatorul SAU și închei jumătatea de integrare cu turul automatizărilor.



50 MIN · DIFICULTATE ●●● · TREI INTRĂRI, O LAMPĂ · AMBELE TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să scrii o condiție cu operatorul SAU și să explici când este adevărată
- Să combini trei intrări (lumină, mișcare, buton) într-o singură decizie
- Să explici ce se evaluează primul într-o condiție cu ȘI și SAU și la ce ajută atunci parantezele
- Să îți prezinți automatizarea în 60 de secunde: prima repetiție pentru Modulul 5

SECȚIUNEA 01 · TEORIE

Operatorul SAU și comanda manuală

DEJA CUNOȘTI (DIN LECȚIA 3.6)
Operatorul ȘI leagă condiții și este adevărat doar când toate sunt adevărate. Becul tău de curte se aprindea la întuneric ȘI mișcare, dar numai atunci.

Fratele promis în 3.6: operatorul SAU (or în Python, || în C++). El este adevărat când cel puțin una dintre condiții este adevărată. Cu el se construiește o regulă de bază a automatizărilor adevărate: comanda manuală; oricât de inteligent este dispozitivul, omul trebuie să poată prelua controlul. Lifturile au buton de deschidere, automobilele au frână de mână, iar lampa ta primește astăzi butonul ei: se aprinde când (este întuneric ȘI este mișcare) SAU când apeși tu. Ordinea în care se evaluează cele două perechi este ca la matematică: ȘI se rezolvă înaintea lui SAU, așa cum înmulțirea se efectuează înaintea adunării. De aceea, parantezele de mai sus nu schimbă rezultatul, dar îl fac limpede dintr-o privire, iar la condiții mai lungi sunt singura ta plasă de siguranță. Le scriem de la bun început, ca pe o regulă de bună practică.

Planul lămpii complete

Schema-bloc: FOTOREZISTOR + PIR + BUTON → „(întuneric ȘI mișcare) SAU buton?” → LED

Pseudocod: repetă: citește lumina, mișcarea și butonul · dacă (lumina sub prag ȘI mișcare) SAU butonul apăsat, aprinde lampa · altfel, stinge-o · pauză scurtă.

SECȚIUNEA 02 · PRACTICĂ

Trei intrări, o decizie

CIRCUIT DE START

▶ Deschide circuitul de start 4.3-A (Micro:bit)

Împărțitorul cu fotorezistor, PIR-ul și LED-ul cu rezistorul lui: exact planșa din 3.6, neconectată și fără cod. Comanda manuală nu se cablează: este chiar butonul A de pe placă. Apasă „Copy and Tinker”.

CIRCUIT DE START

▶ Deschide circuitul de start 4.3-B (Arduino)

Aceleași componente ca în 3.6, plus butonul extern și rezistorul lui de referință de 10 kΩ (din 3.1), toate neconectate și fără cod. Apasă „Copy and Tinker”.

1 · Cablează cele trei intrări și lampa

Planșa este cea din 3.6, cu o singură completare: comanda manuală.

LUMINA · ÎMPĂRȚITORUL DIN 3.3

- alimentare (3V / 5V) → un terminal al fotorezistorului
- celălalt terminal al fotorezistorului → rezistorul de 10 kΩ ← punctul de mijloc
- punctul de mijloc → P0 / A0
- celălalt terminal al rezistorului de 10 kΩ → GND

MIȘCAREA · PIR-UL DIN 3.5

- alimentare (3V / 5V) și GND → alimentarea și GND-ul PIR-ului
- semnalul PIR-ului → P1 / pinul 2

LAMPA

- P2 / pinul 13 → rezistor de protecție → LED → GND

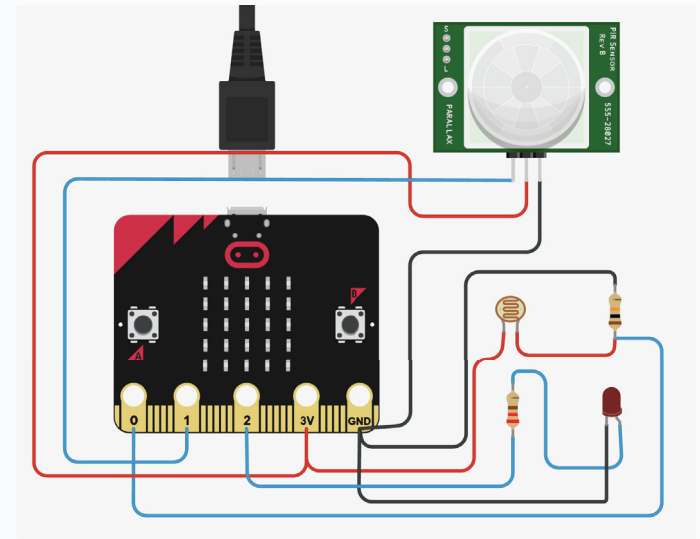
COMANDA MANUALĂ · SINGURA NOUTATE DE PE PLANȘĂ

- Micro:bit: nimic de cablat; folosești butonul A de pe placă, ca în 3.1
- Arduino: un terminal al butonului → 5V; terminalul-pereche opus → pinul 4 și, tot din acel punct, rezistorul de 10 kΩ → GND (schema de intrare din 3.1)

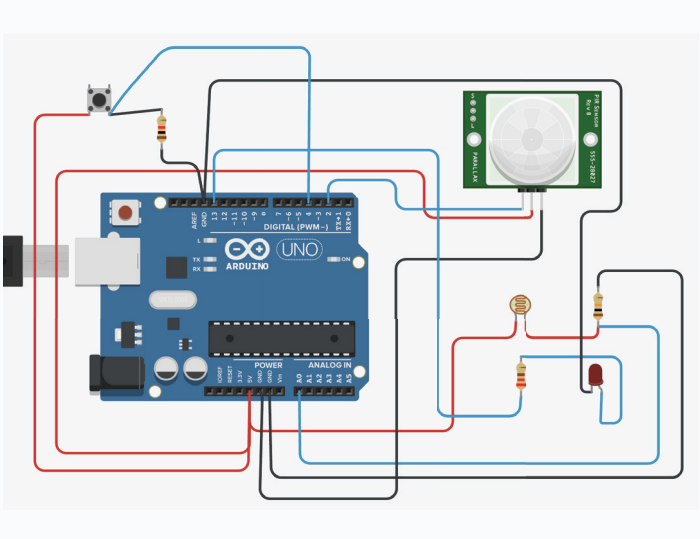
Pinii de alimentare și de GND primesc mai multe fire; este normal: toate componentele se alimentează din aceeași sursă și se întorc pe aceeași masă. Verifică fiecare intrare separat, cu programele din 3.3, 3.5 și 3.1, înainte de a scrie condiția compusă.

☐ Cele trei intrări răspund luate separat

Micro:bit:



Arduino:



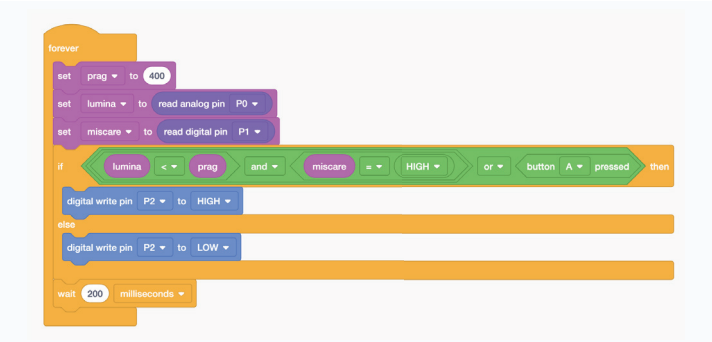
2 · Scrie condiția cu paranteze și verifică toate combinațiile

Micro:bit:
Condiția compusă: (întuneric ȘI mișcare) SAU buton. Cele două semnale din senzori intră mai întâi într-un bloc „and”, care le leagă între ele, iar rezultatul lui se leagă apoi cu butonul printr-un bloc „or”. Blocul din interior se evaluează primul; imbricarea lor arată ordinea la fel de limpede ca parantezele dintr-o formulă. Blocurile compară mișcarea cu HIGH, iar Python-ul cu 1: este același lucru, ca în 4.1. Pragul de lumină din 3.3 îl păstrezi într-o variabilă: în captură, blocul „set prag to 400” este pus în buclă, iar în programul Python de mai jos stă deasupra ei. Funcționează la fel, dar deasupra buclei este locul lui firesc; valoarea nu se schimbă niciodată.

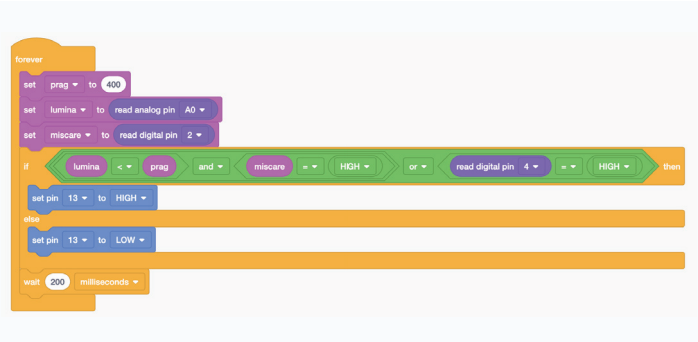
Arduino:
Aceeași logică și aceleași blocuri, cu alte nume de pini. În vitrina C++ din dreapta, ȘI se scrie && și SAU se scrie ||, iar perechea din interior își primește parantezele ei: if ((lumina < PRAG && miscare == HIGH) || buton == HIGH). Parantezele fac în text exact ce fac blocurile imbricate pe planșă: arată dintr-o privire ce se evaluează împreună.

Testează metodic, nu la întâmplare: butonul aprinde lampa și ziua? Întunericul singur, fără mișcare, o lasă stinsă? Mișcarea singură, ziua, o lasă stinsă? Doar întuneric + mișcare o aprinde fără buton?

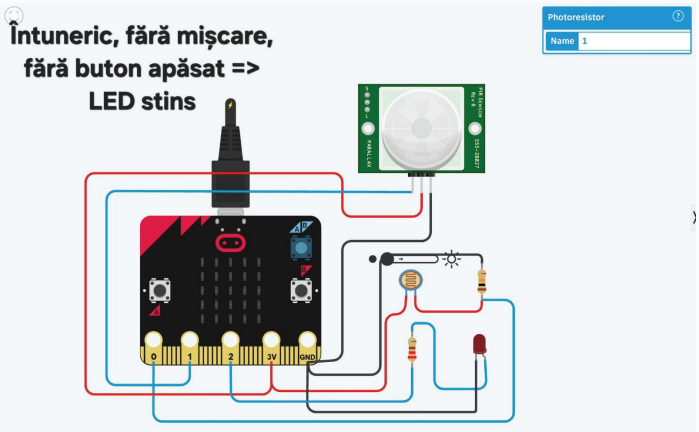
Blocuri de cod · Micro:bit

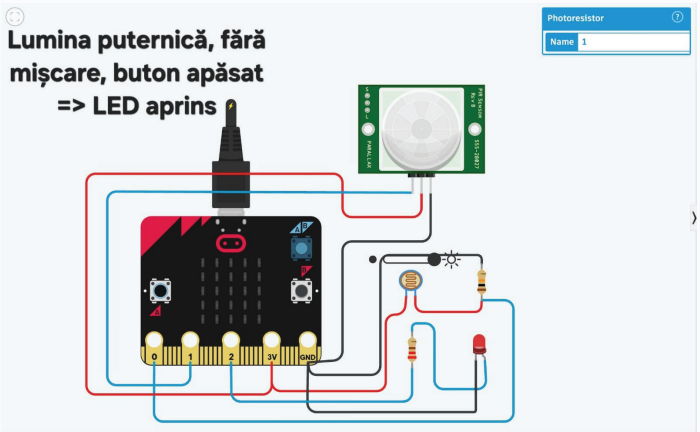


Blocuri de cod · Arduino



Demonstrație pe traseul A (Micro:bit); pe Arduino lampa se comportă identic; diferă doar numele pinilor și butonul, care acolo este cel extern, de pe pinul 4:





☐ Toate combinațiile se poartă corect

TRASEUL A · MICRO:BIT

Lumina → P0 · mișcarea → P1 · lampa pe P2 · comanda manuală: butonul A de pe placă

LAMPA INTELIGENTĂ CU COMANDĂ MANUALĂ

PYTHON · MICRO:BIT

```
# Lecția 4.3: lampa inteligentă cu comandă manuală
# (varianta Tinkercad / MakeCode)

PRAG = 400   # granița de întuneric, din Lecția 3.3

def on_forever():
    lumina = pins.analog_read_pin(AnalogPin.P0)   # intrarea 1: câtă lumină este
    miscare = pins.digital_read_pin(DigitalPin.P1)   # intrarea 2: este mișcare? (1/0)
    buton = input.button_is_pressed(Button.A)   # intrarea 3: comanda manuală
    # parantezele arată ordinea: mai întâi perechea cu ȘI, apoi SAU
    if (lumina < PRAG and miscare == 1) or buton:
        pins.digital_write_pin(DigitalPin.P2, 1)   # atunci: lampa aprinsă
    else:
        pins.digital_write_pin(DigitalPin.P2, 0)   # altfel: stinsă
    basic.pause(200)

basic.forever(on_forever)
```

COD GATA DE COPIAT

▶ Deschide codul Python copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

TRASEUL B · ARDUINO UNO

Lumina → A0 · mișcarea → pinul 2 · lampa pe pinul 13 · butonul extern pe pinul 4

C++ CU COMENTARII ÎN LIMBA ROMÂNĂ — C++ · ARDUINO

```
// C++ code
// Lecția 4.3: lampa inteligentă cu comandă manuală (varianta Arduino)

int PRAG = 400;   // granița de întuneric, din 3.3
int lumina = 0;
int miscare = 0;
int buton = 0;

void setup() {
  pinMode(A0, INPUT);   // fotorezistorul
  pinMode(2, INPUT);   // PIR-ul
  pinMode(4, INPUT);   // butonul manual
  pinMode(13, OUTPUT);   // lampa
}

void loop() {
  lumina = analogRead(A0);   // intrarea 1: câtă lumină este
  miscare = digitalRead(2);   // intrarea 2: este mișcare? (1/0)
  buton = digitalRead(4);   // intrarea 3: comanda manuală
  // && este ȘI, || este SAU; parantezele arată ordinea evaluării
  if ((lumina < PRAG && miscare == HIGH) || buton == HIGH) {
    digitalWrite(13, HIGH);   // atunci: lampa aprinsă
  } else {
    digitalWrite(13, LOW);   // altfel: stinsă
  }
  delay(200);
}
```

COD GATA DE COPIAT

▶ Deschide codul C++ copiabil

Folosește linkul dacă preferi să lipești în loc să scrii.

CARE PLĂCUȚĂ AICI? · AVANTAJE ȘI DEZAVANTAJE ÎN SITUAȚIA DE AZI

MICRO:BIT

Butonul manual este deja pe placă, iar condiția cu paranteze este o singură propoziție Python, lizibilă de la un capăt la altul.

ARDUINO UNO

Butonul se cablează, dar blocurile and/or imbricate unele în altele arată vizual exact ordinea evaluării; ideal pentru prima înțelegere.

Concluzia: condițiile compuse se învață bine din blocuri și se scriu repede în text. Iar comanda manuală nu este opțională: orice automatizare serioasă o are.

SECȚIUNEA 03 · DEPANARE

⚠ Logica s-a încurcat?

1) Lampa se aprinde ziua, fără buton → nu parantezele sunt de vină (ȘI se evaluează oricum înaintea lui SAU): verifică pragul de lumină din 3.3 și, la Arduino, rezistorul de referință al butonului. 2) Ai grupat altfel decât ai intenționat, de exemplu întuneric ȘI (mișcare SAU buton) → lampa nu se mai aprinde ziua la apăsarea butonului; aici gruparea chiar schimbă rezultatul, așa că citește condiția cu voce tare înainte de a o scrie. 3) Butonul nu are efect → la Arduino, verifică rezistorul de referință de pe pinul 4 (schema din 3.1); fără el, pinul „plutește” și citește la întâmplare. 4) Lampa rămâne aprinsă după ce mișcarea a trecut → este normal câteva secunde: PIR-ul menține semnalul ridicat o vreme (3.5). 5) Testul complet are 8 combinații (2×2×2); bifează-le pe caiet într-un tabel: lumină / mișcare / buton → lampă. Tabelul acesta are un nume oficial: tabel de adevăr.

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (10 minute, în perechi)

Finalul jumătății de integrare se sărbătorește ca la o expoziție: fiecare pereche își alege cel mai reușit dispozitiv din 4.1-4.3 (sau o variantă din provocări) și îl prezintă în exact 60 de secunde, după o structură în trei propoziții: CE face („bariera se ridică singură...”) · CUM decide („...când distanța scade sub 20 cm...”) · CE ar îmbunătăți („...i-am adăuga coborârea în trepte”). Clasa cronometrează și pune o singură întrebare per echipă. Este prima repetiție, în miniatură, pentru prezentarea proiectului din Modulul 5; acolo structura crește, dar scheletul rămâne același.

Varianta Kahoot: tabele de adevăr de completat pentru condiții cu ȘI/SAU; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 4.3-J: tabelul de adevăr al lămpii (8 rânduri), plus structura prezentării de 60 de secunde, de completat
Varianta fără ecrane: chiar jocul principal; necesită un cronometru și curajul de a vorbi în fața clasei

Provocare: lampa de serie mare

Nivel 1 · Veghea. La întuneric fără mișcare, lampa nu stă stinsă de tot: luminează la 20 % din scara plăcii tale, cu scrierea analogică din 2.5. Pe Arduino, mută mai întâi LED-ul pe un pin cu ~ (de exemplu 9); pinul 13 nu poate genera PWM.

Nivel 2 · A treia condiție. Provocarea din 3.6 se întoarce: adaugă și temperatura în decizie. Poți combina ȘI cu SAU? Da, cu paranteze la fiecare pereche.

3 întrebări rapide

1. Când este adevărată o condiție legată cu SAU?
2. Într-o condiție cu ȘI și SAU, care se evaluează primul și la ce ajută atunci parantezele?
3. Ce este comanda manuală și de ce o are orice automatizare serioasă?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

operatorul SAU (or / ||): leagă condiții; adevărat când cel puțin una este adevărată

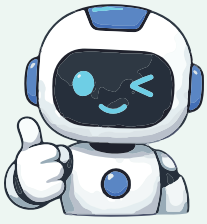
paranteze în condiții: grupează perechile și arată intenția; ordinea o dă oricum ȘI, evaluat înaintea lui SAU

comandă manuală: calea prin care omul preia oricând controlul automatizării

tabel de adevăr: tabelul tuturor combinațiilor de intrări și al rezultatului fiecăreia

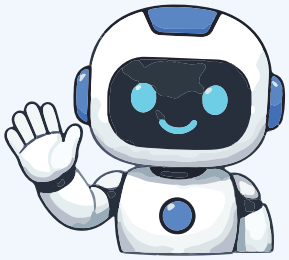
Ce trebuie să știi după această lecție

- ✓ ȘI cere toate condițiile; SAU se mulțumește cu una, iar ȘI se evaluează înaintea lui SAU, ca înmulțirea înaintea adunării
- ✓ Parantezele nu schimbă rezultatul aici, dar arată intenția: le scrii ca să te citești pe tine
- ✓ Trei intrări = 8 combinații; testul complet le bifează pe toate în tabelul de adevăr
- ✓ Comanda manuală este regulă de proiectare, nu un moft: omul preia oricând controlul
- ✓ Structura prezentării de 60 de secunde: CE face · CUM decide · CE ai îmbunătăți



LECȚIA 4.4 · 50 DE MINUTE

Cum funcționează o imprimantă 3D



De astăzi schimbăm instrumentele: din lumea circuitelor trecem în lumea obiectelor. Afli de unde vin imprimantele 3D, cum construiesc ele orice formă strat cu strat și cine le mișcă: un vechi cunoscut din Modulul 2.

50 MIN · DIFICULTATE ●○○ · TEORIE + CALCUL · FĂRĂ TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să explici diferența dintre fabricarea prin adăugare și cea prin îndepărtare de material
- Să descrii, pas cu pas, cum construiește o imprimantă FDM un obiect
- Să calculezi câte straturi are o piesă, din înălțimea ei și înălțimea stratului

SECȚIUNEA 01 · TEORIE

Scurtă istorie: de la laborator la birou

DEJA CUNOȘTI (DIN LECȚIA 2.6)

Caseta-vitrină de atunci: motorul pas cu pas, care se rotește în pași ficși de 1,8°. Ți-am promis că îl revezi când imprimanta desenează straturi perfecte. Astăzi este ziua aceea.

Prima imprimantă 3D nu topea plastic: în 1983, inginerul american Chuck Hull a întărit cu lumină, strat după strat, o rășină lichidă, procedeu numit stereolitografie, brevetat în 1986. Câțiva ani mai târziu, Scott Crump a inventat metoda care domină astăzi birourile și școlile: FDM (modelarea prin depunere de material topit). Zeci de ani, ambele au rămas instrumente industriale de zeci de mii de dolari. Cotitura a venit în 2005, cu proiectul RepRap al profesorului Adrian Bowyer: o imprimantă cu planuri publicate gratuit, gândită să își poată tipări singură o parte din propriile piese. De atunci, prețurile au scăzut dramatic, iar astăzi o imprimantă FDM decentă costă cât un telefon.

Principiul: orice formă, felie cu felie

Sculptorul lucrează prin îndepărtare: pleacă de la un bloc și taie ce este în plus. Imprimanta 3D lucrează exact invers, prin adăugare: pleacă de la nimic și depune material doar acolo unde trebuie, un strat subțire peste altul. De aceea procedeul se numește fabricare aditivă. Puterea metodei stă într-o idee simplă: orice formă, oricât de complicată, devine ușoară dacă o tai în felii subțiri: fiecare felie este doar un desen plat, iar desene plate poate realiza orice mașină cu două motoare.

Piesă nouă: Imprimanta FDM

PRIMA APARIȚIE

CE ESTE

Un dispozitiv cu un cap de imprimare încălzit care se deplasează deasupra unei mese. FDM vine din limba engleză: Fused Deposition Modeling, adică modelare prin depunerea materialului topit.

CE FACE

Trage filamentul de plastic în capul încălzit la circa 200 °C, îl topește și îl împinge printr-o duză subțire (de obicei 0,4 mm), desenând conturul și umplutura fiecărui strat. Când stratul este gata, capul urcă tipic 0,2 mm și desenează următorul: sute de straturi, ore întregi, milimetru cu milimetru.

CINE O MIȘCĂ

Trei motoare pas cu pas, vechiul cunoscut din 2.6: pași ficși de 1,8°, transformați prin curele și șuruburi în deplasări de sutimi de milimetru pe axele X, Y și Z. Precizia obiectului este precizia pașilor lor.

UNDE O ÎNTÂLNEȘTI

Școli, biblioteci, ateliere de reparații, cabinete de tehnică dentară, firme care fac prototipuri, precum și tot mai multe camere de elevi.

Piesă nouă: Filamentul PLA

PRIMA APARIȚIE

CE ESTE

„Cerneala” imprimantei: un fir de plastic cu diametrul de 1,75 mm, înfășurat pe o bobină. PLA (acid polilactic) este plasticul școlilor: fabricat din amidon de porumb, aproape fără miros la tipărire.

CE FACE

Se topește ușor (în jur de 200 °C), se lipește strat de strat și se întărește imediat ce iese din duză. Rezultatul este o piesă rigidă, cu striațiile fine ale straturilor vizibile pe margine.

RUDELE LUI

PETG: mai flexibil și mai rezistent la soare, pentru piese de exterior; ABS: plasticul jucăriilor clasice, rezistent, dar pretențios la tipărire. Alegerea materialului este o decizie de proiectare, ca alegerea senzorului.

DE CE NE TREBUIE

Piesele robotului vostru din Modulul 5 (suporturi de senzori, brațe de barieră, carcase) se vor tipări din el.

Matematica lecției: numărul de straturi

numărul de straturi = înălțimea piesei ÷ înălțimea unui strat

O piesă de 3 cm tipărită cu straturi de 0,2 mm: 30 ÷ 0,2 = 150 de straturi. Aceeași piesă la 0,1 mm: 300 de straturi, de două ori mai fină, dar de aproape două ori mai lentă. Primul compromis al imprimării 3D: calitate contra timp. Îl vei regla cu mâna ta în Lecția 4.6.

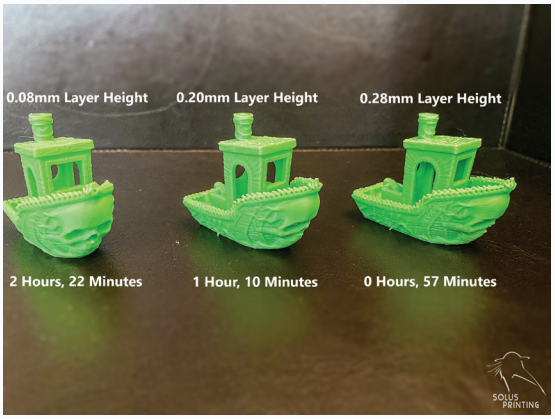


Fig. 1. Aceeași piesă, tipărită cu trei înălțimi de strat (0,08 · 0,20 · 0,28 mm). Valorile nu sunt exact cele din lecție, dar regula se vede limpede: cu cât stratul este mai subțire, cu atât piesa este mai fină și tipărirea durează mai mult.

SECȚIUNEA 02 · EXERSĂM

Rezolvă pe caiet, cu formula în față.

1. Un breloc de 4 mm grosime, tipărit cu straturi de 0,2 mm: câte straturi are?

2. O figurină de 8 cm a ieșit din imprimantă cu 400 de straturi. Cu ce înălțime de strat a fost tipărită?

3. Un pod în miniatură are o arcadă suspendată în aer. Ce problemă are imprimanta la straturile arcadei și ce soluție bănuiești? (Răspunsul oficial vine în 4.6.)

SECȚIUNEA 03 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, pe echipe)

Un elev este capul de imprimare: stă la tablă cu creta pe punctul de start și execută doar comenzi de deplasare primite de la echipa lui („dreapta 20 cm”, „sus 10”, „stânga 20”, „jos 10”), desenând conturul unei forme secrete primite de echipă pe un bilet (un L, un T, o treaptă). Restul clasei ghicește forma cât mai devreme. Regula de aur, identică cu a robotului din 1.6: capul face doar ce i se spune, exact în ordinea spusă; o comandă uitată și conturul rămâne deschis. Runda a doua: aceeași formă, „strat cu strat”; conturul se retrasează de 3 ori, puțin mai sus, pentru a se simți fizic răbdarea imprimării.

Varianta Kahoot: istoria și principiul în 10 întrebări; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 4.4-J: forme de „feliat”; desenează cele 3 straturi de jos ale fiecărei piese date
Varianta fără ecrane: chiar jocul principal; necesită tablă, cretă și comenzi stricte

Provocare: arheologia din cameră

Găsește acasă sau în școală un obiect tipărit 3D (multe carcase, suporturi de telefon și jucării de târg sunt tipărite astfel). Privește-l în lumină razantă: vezi striațiile? Numără-le pe 1 cm și estimează înălțimea stratului cu formula aplicată invers. Adu concluzia la ora următoare.

3 întrebări rapide

1. Prin ce se deosebește fabricarea aditivă de cea prin îndepărtare?
2. De ce construiește imprimanta în straturi, și nu obiectul dintr-o bucată?
3. Ce rol joacă motoarele pas cu pas din 2.6 în precizia imprimantei?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

fabricare aditivă: construirea obiectului prin adăugare de material, strat cu strat

FDM: metoda cu plastic topit depus prin duză; imprimanta de birou și de școală

duză: vârful subțire (tipic 0,4 mm) prin care iese firul de plastic topit

strat: o „felie” a obiectului, tipic de 0,2 mm înălțime

filament: firul de plastic de pe bobină; „cerneala” imprimantei

PLA: acid polilactic; plasticul școlilor: din amidon de porumb, ușor de tipărit

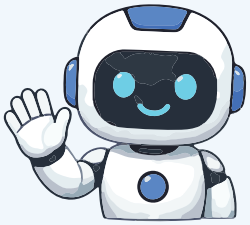
Ce trebuie să știi după această lecție

- ✓ Imprimanta 3D adaugă material doar unde trebuie; invers față de sculptor
- ✓ Orice formă devine simplă tăiată în felii; fiecare strat este un desen plat
- ✓ FDM: filament topit la ~200 °C, duză de 0,4 mm, straturi de ~0,2 mm, mișcate de motoare pas cu pas
- ✓ Numărul de straturi = înălțimea piesei ÷ înălțimea stratului; primul compromis: calitate contra timp

LECȚIA 4.5 · 50 DE MINUTE

Primii pași în Tinkercad 3D

Același cont, alt atelier: din Circuits treci în 3D Designs. Înveți cele trei operații din care se naște orice model (forme, găuri, grupare) și pleci acasă cu primul tău obiect: un breloc cu numele tău.



50 MIN · DIFICULTATE ●●○ · PROIECTARE 3D · FĂRĂ TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să te orientezi în atelierul 3D: planul de lucru, formele, vederea din orice unghi
- Să folosești perechea solid + gaură și gruparea: cele trei operații ale proiectării
- Să construiești brelocul cu numele tău, la dimensiuni reale, în milimetri

SECȚIUNEA 01 · TEORIE

Atelierul 3D și cele trei operații

DEJA CUNOȘTI (DIN LECȚIA 4.4)

Imprimanta construiește orice formă din felii. Forma trebuie însă mai întâi desenată, în trei dimensiuni și la dimensiuni reale. Această activitate se numește proiectare asistată de calculator (pe scurt CAD, din limba engleză: Computer-Aided Design).

Deschide tinkercad.com cu contul tău obișnuit. În tabloul de bord, în bara din stânga, alege 3D Designs în locul secțiunii Circuits, apoi apasă Create → 3D Design. Te întâmpină planul de lucru: o masă albastră cu caroiaj, gradată în milimetri reali; un cub de 20 pe ecran va avea exact 20 de milimetri și în palmă, după tipărire. În dreapta se află dulapul cu forme de bază: cutii, cilindri, sfere, piramide, text. Vederea o rotești în orice direcție (ținând apăsat butonul drept al mouse-ului sau folosind cubul de navigare din colțul din stânga sus); obișnuiește-te să îți privești piesa din toate părțile, ca un olar.

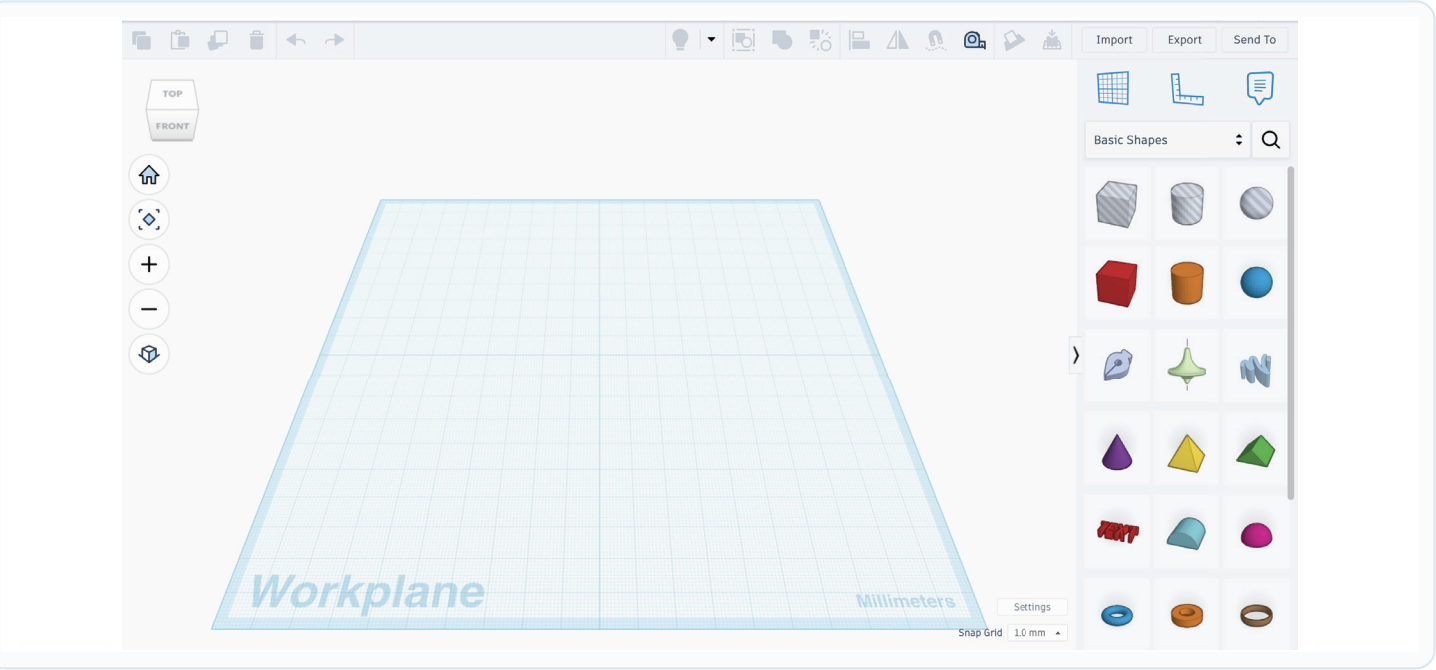


Fig. 1. Atelierul 3D: în centru planul de lucru gradat în milimetri, în dreapta dulapul cu forme de bază, în stânga sus cubul de navigare cu care rotești vederea.

Cele trei operații din care se naște orice model

1 Aduci și potrivești forme

Tragi o formă pe plan, apoi o redimensionezi din mânerele albe și scrii dimensiunile exacte în casetele de milimetri; la proiectare nu se lucrează „din ochi”.

2 Scobești cu găuri

Orice formă poate deveni „gaură” (Hole) dintr-un clic: acolo unde gaura se suprapune cu un solid, materialul dispare. Un cilindru-gaură printr-o cutie = un orificiu perfect.

3 Grupezi

Selectezi piesele și apeși Group: formele și găurile devin un singur obiect. Din aceste trei operații, repetate cu răbdare, sunt construite toate modelele din lume, de la brelocul de astăzi la carcasa robotului vostru.

SECȚIUNEA 02 · PRACTICĂ

Primul tău obiect: brelocul cu nume

PROIECT DE START

▶ Deschide proiectul de start 4.5 · Brelocul

Un plan de lucru gol, cu brelocul-exemplu terminat așezat alături, ca reper de mărime. Apasă „Copy and Tinker” (în 3D Designs butonul are aceeași denumire) și construiește lângă exemplu, nu peste el.

1 · Baza brelocului

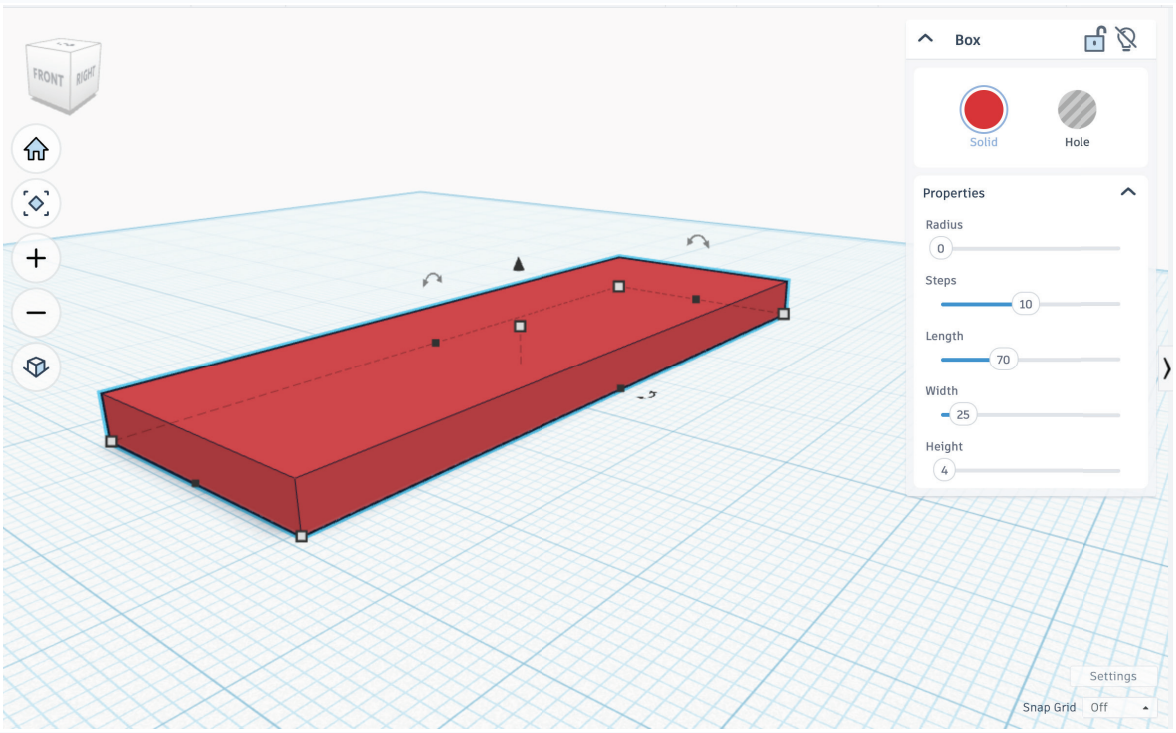
Din dulapul din dreapta (Basic Shapes) trage o cutie (Box) pe planul de lucru. Cu ea selectată se întâmplă două lucruri: pe piesă apar mânerele albe, iar în dreapta-sus se deschide panoul de proprietăți al formei.

Cotele se scriu, nu se trag de mânere. La cutie le găsești direct în panoul din dreapta: Length (lungime), Width (lățime), Height (înălțime). Dă click în fiecare căsuță și scrie exact 70, 25 și 4, în milimetri.

Ține minte și cealaltă cale, pentru că la alte forme e singura: dacă dai click pe un mâner alb de pe planul de lucru, chiar lângă el apare o căsuță numerică în care poți scrie cota. Panoul din dreapta arată altceva la fiecare formă; căsuțele de lângă mânere sunt mereu acolo.

Verifică și poziția pe verticală: baza trebuie să stea PE plan, adică la înălțimea 0. Deasupra piesei, în vârf, e un mâner negru în formă de con: el ridică forma deasupra planului. Dă un click pe el și apare o căsuță cu înălțimea curentă — dacă nu scrie 0, scrie tu 0 și apasă Enter. Rotește vederea și privește piesa din lateral: trebuie să stea lipită de plan, fără umbră de aer sub ea.

Patru milimetri grosime înseamnă 20 de straturi la imprimare — formula din lecția trecută, aplicată pentru prima dată pe o piesă a ta.



- [ ] Baza are exact 70 × 25 × 4 mm și stă pe plan

2 · Numele tău, din forma Text

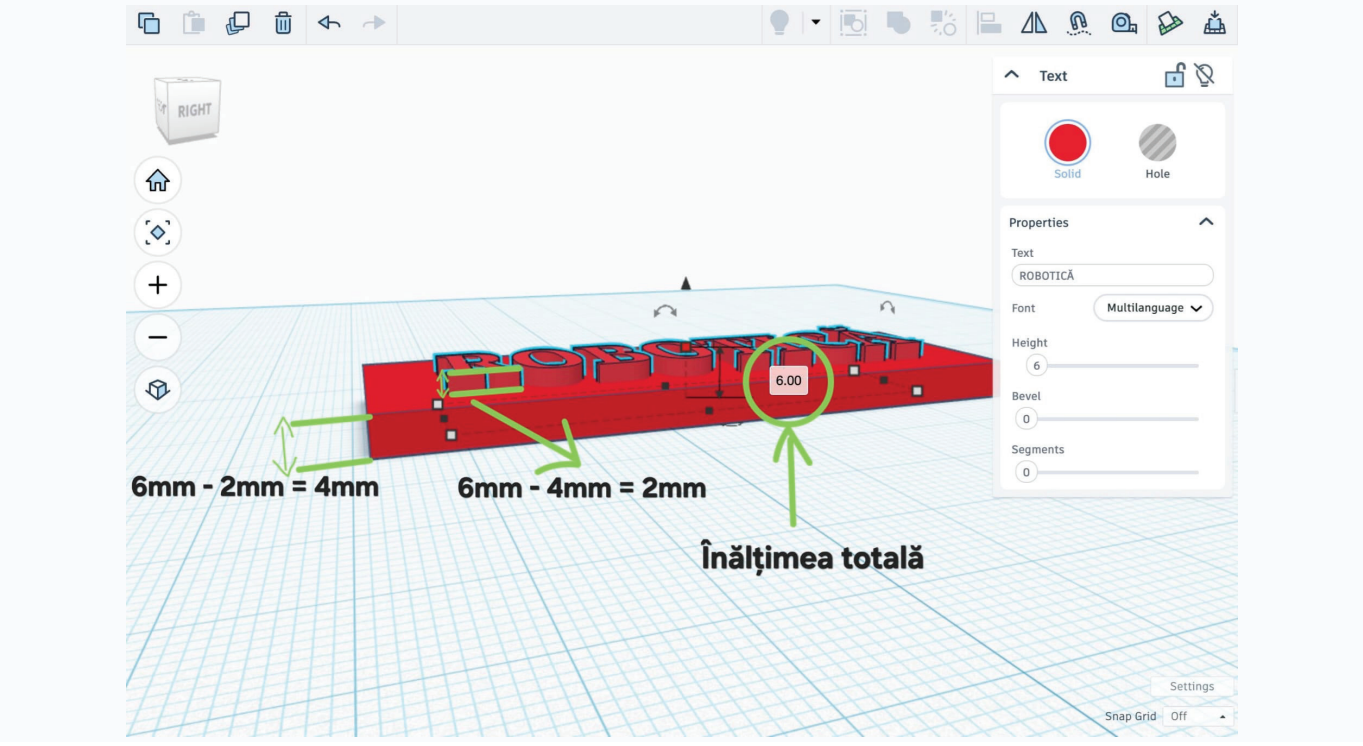
Caută în dulapul din dreapta forma “Text” și trage-o pe plan. În căsuța de proprietăți care se deschide, șterge cuvântul implicit și scrie-ți numele; tot de acolo poți schimba fontul — lasă-l deocamdată cum e.

Acum dimensiunile. Panoul din dreapta al formei Text are doar Height — grosimea pe verticală: scrie acolo 6. Lungimea și lățimea se dau din căsuțele de lângă mânerele albe, de pe planul de lucru: click pe mânerul alb din colț, scrie 55 în căsuța de pe lungime și 14 în cea de pe lățime. Pentru un nume de 5–7 litere, atât încape frumos pe bază; dacă numele tău e mai lung, mărește lungimea până când literele stau tot pe bază, fără să atârne peste muchie.

Aici se greșește cel mai des: textul rămâne la înălțimea 0, sprijinit pe plan. Nu-l ridici deasupra bazei. Baza are 4 mm, litera 6 — deci 4 mm intră în bază, 2 ies în relief.Așa litera și baza-s un singur corp și nu se rupe. Ridicat la 4 mm, textul doar atinge baza — contact pe o linie, piesa iese fragilă.

Pentru centrare, selectează ambele piese (click pe una, Shift + click pe cealaltă) și apasă Align (sus, în bara de comenzi): apar mânerele de aliniere pe cele trei direcții; alege mijlocul pe lățime. Pe lungime NU centra: capătul din stânga rămâne liber pentru gaura de inel, pe care o faci la pasul 3. Mută textul spre dreapta până când între prima literă și muchia scurtă din stânga rămân vreo 12 mm. Nu alinia nici pe vertical — ai strica înălțimile pe care tocmai le-ai reglat. În capturile lecției am scris „ROBOTICĂ” — opt litere, la 62 mm lungime. E același drum: scrii ce vrei, apoi potrivești lungimea până când textul stă tot pe bază.

2 · Numele tău, din forma Text · verificarea geometriei



Align: centrăm pe lățime, lăsăm loc pentru gaura de inel

- [ ] Numele stă centrat, cu 2 mm de relief peste bază

3 · Gaura pentru inel și gruparea

Trage un cilindru (Cylinder) pe plan. Dă-i din căsuțe diametrul de 4 mm pe ambele laturi și înălțimea de 10 mm — mai înalt decât piesa, ca să o străpungă sigur dintr-o parte în alta.

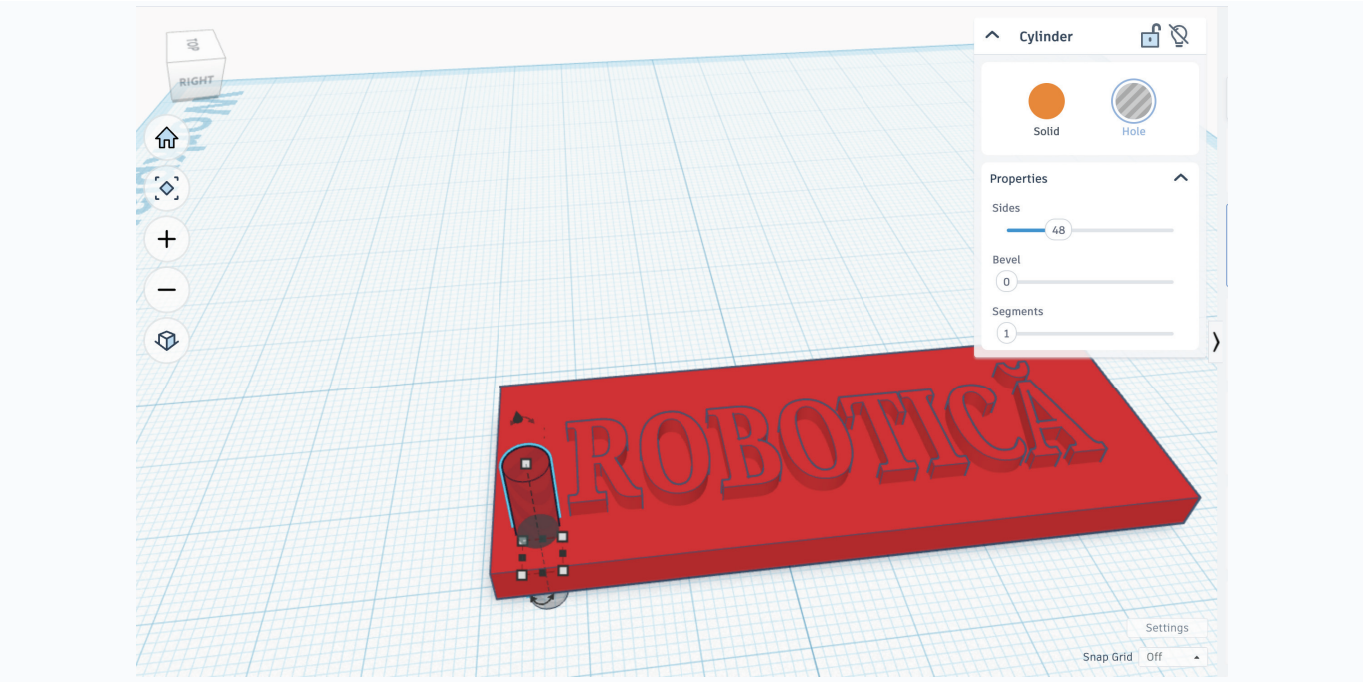
Cu cilindrul selectat, apasă butonul Hole din panoul din dreapta: devine gri-transparent, semn că de acum scoate material în loc să adauge.

Așază-l în capătul din stânga al bazei, la aproximativ 6 mm de muchia scurtă și pe mijlocul lățimii. În jurul găurii trebuie să rămână material: o gaură lipită de muchie se rupe la prima cheie agățată de ea. Trage cilindrul cu mouse-ul până aproape de loc, apoi ajustează fin cu săgețile de pe tastatură — fiecare apăsare îl mută cu un pas mic, iar cu Shift apăsat se mută mai mult odată. Dacă vrei pași ficși, pune Snap Grid (dreapta-jos) pe 1 mm. Mai rămâne un lucru: cilindrul stă acum la înălțimea 0, adică pornește chiar de pe plan. Dă click pe mânerul negru din vârf și scrie −1: coboară cu un milimetru sub plan. Rotește apoi vederea și verifică din lateral că trece prin toată grosimea bazei, ieșind și pe deasupra, și pe dedesubt.

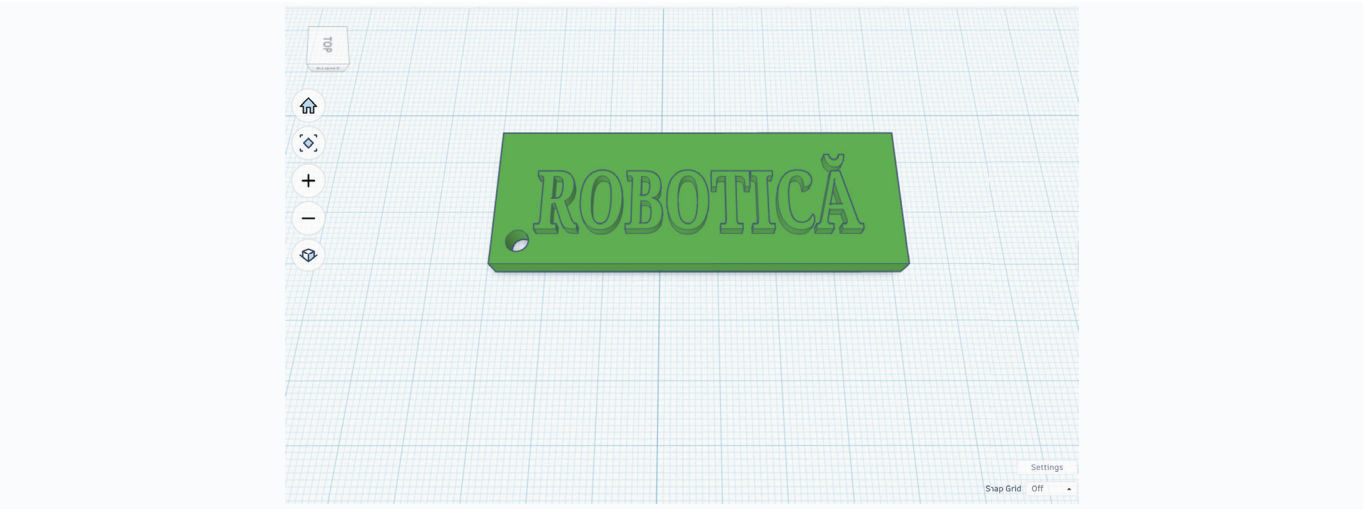
Selectează tot (Ctrl+A) și apasă Group, sau Ctrl+G: baza, literele și gaura devin un singur obiect. Abia acum gaura scobește cu adevărat. Dacă la grupare piesa devine toată de o culoare, e normal — alege-i alta din meniul Solid.

1. **SOLID**
2. **HOLE**
3. **GROUP**

Gaura înainte de Group



Obiectul grupat



La fel ca în cazul circuitelor, nu trebuie să salvezi nimic: Tinkercad salvează automat, la fiecare modificare. Pentru a-ți regăsi brelocul data viitoare, execută clic pe „Tinkercad” din colțul din stânga sus și intră în 3D Designs : proiectul se află acolo, cu numele generat automat. Îl poți redenumi „Breloc”, ca să îl recunoști: în Lecția 4.6 ai nevoie exact de el.

- ☐ Brelocul e un singur obiect, cu gaura străpunsă dintr-o parte în alta

Pentru avansați: aceleași forme, din cod · Tinkercad Codeblocks

În meniul Tinkercad există și secțiunea Codeblocks: acolo formele nu se trag cu mouse-ul, ci se apelează din blocuri de cod, ca cele din Circuits: „adaugă cutie, mut-o cu 20, rotește-o cu 45°”. Programare și proiectare, în aceeași activitate. Provocarea pentru cei rapizi: reconstruiește-ți brelocul numai din blocuri. Nu este materie obligatorie, ci o ușă deschisă.

☐ SECȚIUNEA 03 · DEPANARE

⚠ Modelul nu arată ca în plan?

1) Gaura nu a scobit nimic → ai grupat? Găurile acționează abia la Group. 2) Gaura nu străpunge → cilindrul este prea scund sau nu este poziționat corect pe grosime; fă-l mai înalt decât piesa și lasă-l să iasă și pe deasupra, și pe dedesubt. 3) Textul plutește deasupra bazei sau este complet îngropat → privește piesa din lateral (rotește vederea!) și corectează înălțimea din caseta ei, nu trăgând de săgeata verticală. 4) Ai șters ceva din greșeală → Ctrl+Z. 5) Piesele se deplasează împreună, deși nu vrei → le-ai grupat prea devreme; Ungroup le desparte oricând.

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (6 minute, în perechi)

Ochiul de proiectant vede lumea în forme de bază, iar jocul antrenează exact această deprindere. Un elev primește pe bilet un obiect obișnuit (cană, creion, casă cu acoperiș, clepsidră, haltere) și îl descrie DOAR în limbaj de atelier: „un cilindru înalt, gol în interior, adică un cilindru-gaură mai subțire în el, plus o jumătate de tor pe lateral”. Perechea desenează pe caiet ce aude și ghicește obiectul. Apoi rolurile se schimbă. Descoperirea care rămâne: aproape orice obiect din jur este o sumă de 3-4 forme de bază și 1-2 găuri.

Varianta Kahoot: obiecte de recunoscut din descompunerea lor în forme; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 4.5-J: patru obiecte desenate; scrie descompunerea în forme a fiecăruia (formele + găurile + gruparea)
Varianta fără ecrane: chiar jocul principal; necesită bilete, caiete și vocabularul formelor

Provocare: piesa care contează

Brelocul a fost exercițiul de încălzire. Adevărata țintă: o piesă pentru robotul din Modulul 5, adică un suport de senzor ultrasonic (o cutie mică cu două găuri rotunde pentru „ochii” senzorului) sau un braț de barieră cu gaură de prindere pe cornul servomotorului. Dimensiunile reale ale componentelor le găsești în fișele lor tehnice; începe cu schița pe caiet, în milimetri.

3 întrebări rapide

1. Care sunt cele trei operații din care se construiește orice model?
2. Ce se întâmplă cu o formă-gaură la momentul grupării?
3. De ce se scriu dimensiunile în casete, în loc să tragi „din ochi” de mânere?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

proiectare (CAD): desenarea obiectului în 3D, la dimensiuni reale, înainte ca el să existe fizic

plan de lucru: masa gradată în milimetri pe care construiești modelul

solid / gaură (Hole): forma care adaugă material / forma care îl înlătură

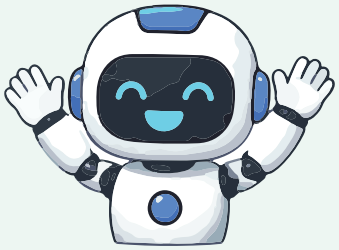
grupare (Group): unirea formelor și găurilor într-un singur obiect

aliniere (Align): instrumentul care centrează piesele exact una față de alta

Codeblocks: atelierul unde formele 3D se construiesc din blocuri de cod

Ce trebuie să știi după această lecție

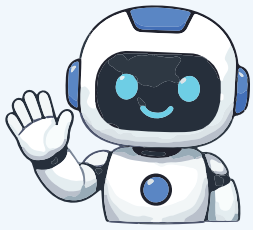
- ✓ Trei operații: aduci și potrivești forme · scobești cu găuri · grupezi
- ✓ Milimetrii din casete sunt milimetri reali: proiectezi la scară 1:1
- ✓ Găurile scobesc abia la Group; Ungroup desface oricând
- ✓ Privește piesa din toate unghiurile: vederea rotită este ochiul proiectantului



LECȚIA 4.6 · 50 DE MINUTE

De la model la obiect: STL și slicing

Ultima verigă a lanțului: modelul tău pleacă din Tinkercad ca fișier STL, iar un program numit slicer îl taie în straturi și scrie instrucțiunile imprimantei. Închei modulul cunoscând tot drumul: de la idee la obiect.



50 MIN · DIFICULTATE ●●○ · EXPORT + DEMONSTRAȚIE DE SLICING · FĂRĂ TRASEE

CE VEI ȘTI LA FINALUL LECȚIEI

- Să exporți modelul tău din Tinkercad ca fișier STL
- Să explici ce face un slicer și ce înseamnă cele trei setări-cheie: stratul, umplerea, suporturile
- Să alegi setări potrivite pentru o piesă dată și să argumentezi alegerea

SECȚIUNEA 01 · TEORIE

STL: modelul, împachetat pentru drum

DEJA CUNOȘTI (DIN LECȚIILE 4.4 ȘI 4.5)

Imprimanta lucrează în straturi, iar modelul tău este gata în Tinkercad. Între ele mai lipsesc două verigi: un fișier de transport și un traducător.

Fișierul de transport se numește STL, formatul universal al imprimării 3D, citit de orice slicer din lume. El descrie doar suprafața modelului, acoperită cu o rețea de triunghiuri minuscule: mii de triunghiuri care, împreună, îmbracă exact forma ta. Exportul din Tinkercad se face în trei clicuri: butonul Export din dreapta sus → ce anume exporți (tot modelul sau doar selecția) → formatul .STL. Atât, iar modelul tău poate ajunge la orice imprimantă din lume.

Slicerul: traducătorul în limba imprimantei

Al doilea program din lanț este slicerul , adică „feliatorul”. Noi îl prezentăm pe Bambu Studio, un slicer gratuit folosit frecvent în școli: primește STL-ul, îl așază pe masa virtuală a imprimantei, îl taie în straturi după setările alese și scrie rezultatul sub formă de G-code: lista de instrucțiuni pe care imprimanta le execută în ordine, exact cum au fost scrise. Sună cunoscut? Este, cuvânt cu cuvânt, definiția programului din Lecția 1.1, doar că acest program nu îl scrii tu, ci îl generează slicerul: zeci de mii de rânduri pentru un breloc, sute de mii pentru o piesă complicată.

▶ Scanează codul QR și descarcă Bambu Studio

Bambu Studio este slicerul gratuit oferit de Bambu Lab (producător de imprimante 3D). Scanează codul QR sau accesează "https://bambulab.com/en-eu/download/studio" pentru a instala Bambu Studio.

⚠ Important: partea de slicing e demonstrativă

Exportul STL îl poate face oricine: se realizează în Tinkercad, din contul tău. Pașii de slicing au însă nevoie de un slicer instalat și, pentru a avea sens, de o imprimantă: îi urmărești ca demonstrație (la ecran, cu profesorul) și îi parcurgi tu doar dacă ai acces la o imprimantă, cu Bambu Studio sau cu orice alt slicer. Nimic din evaluarea manualului nu depinde de existența unei imprimante acasă.

Cele trei setări pe care le atinge oricine

Înălțimea stratului

Felia din 4.4; 0,2 mm este valoarea standard, 0,1 mm pentru finețe (timp dublu), 0,3 mm pentru rapiditate.

Umplerea (infill)

Piesele nu se tipăresc pline; în interior primesc o structură de tip fagure, rară, tipic 15 %. Astfel se economisește plastic și timp, iar piesa rămâne surprinzător de rezistentă. Piesele solicitate mecanic (un braț de barieră) urcă spre 30-40 %.

Suporturile

Răspunsul la exercițiul 3 din Lecția 4.4; plasticul topit nu poate fi depus în aer. Porțiunile înclinate cu mai mult de aproximativ 45° față de verticală primesc sub ele schele subțiri, tipărite odată cu piesa și îndepărtate la final. În slicer, unghiul este măsurat invers, față de masă, nu față de verticală („Threshold angle"): cele două unghiuri sunt complementare, se completează până la 90°, deci regula noastră de 45° față de verticală înseamnă tot 45° față de masă. Valoarea implicită a slicerului, 30°, este mai permisivă: generează schele doar sub porțiunile și mai apropiate de orizontală. Cea mai bună soluție rămâne însă proiectarea inteligentă: piese care nu au nevoie deloc de schele.

SECȚIUNEA 02 · PRACTICĂ

Drumul complet, cu brelocul tău

▶ Deschide proiectul 4.5 · Brelocul

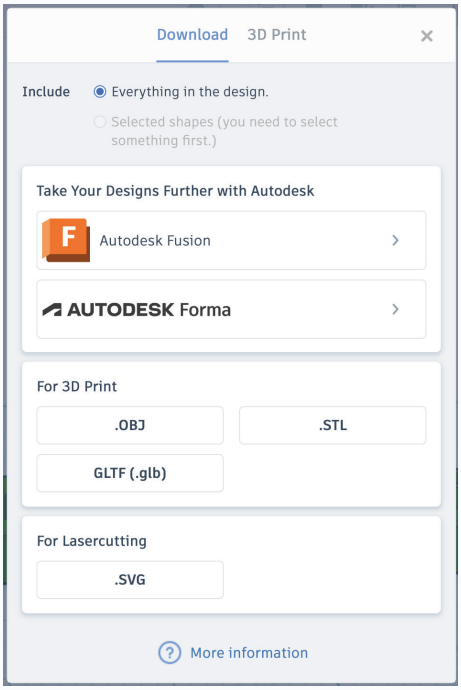
Dacă nu ai finalizat proiectul lecției trecute, îl poți prelua de aici. Apasă „Copy and Tinker”.

1 · Exportă STL-ul (toată lumea)

Deschide brelocul din lecția trecută (din 3D Designs, în contul tău). Pentru fișierul întregii piese nu selecta nimic pe plan; execută un clic pe fundal, ca să te asiguri că nu ai nimic selectat. Dacă dorești doar o parte din model, selecteaz-o mai întâi; altfel, opțiunea „Selected shapes” rămâne inactivă și nu poate fi bifată.

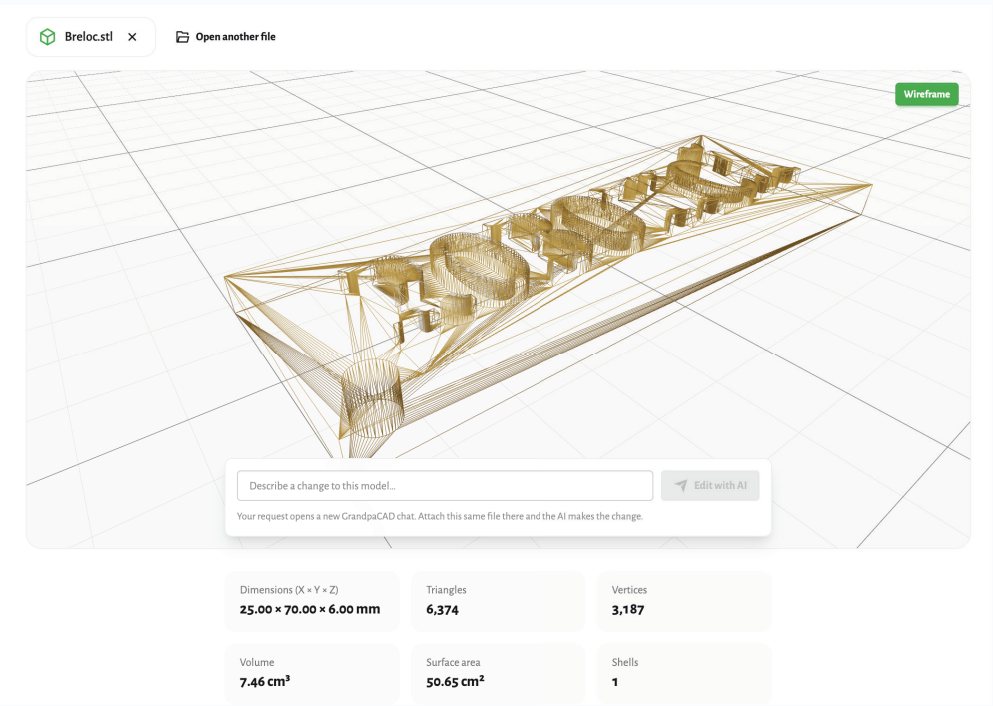
Apasă butonul Export din dreapta sus. Se deschide o fereastră numită Download, cu două file în partea de sus; rămâi pe Download, cea activă. Alegi mai întâi ce exporți, din dreptul cuvântului Include: „Everything in the design” (tot modelul) sau „Selected shapes” (doar formele selectate; opțiunea este inactivă dacă nu ai selectat nimic). Abia apoi, la For 3D Print, apeși .STL. Fișierul se descarcă automat în folderul Descărcări.

Fereastra Export · .STL



Privește-l acum în interior. Deschide fișierul cu un vizualizator de STL din browser (noi am folosit „STL Viewer” de pe grandpacad.com, care afișează și câte triunghiuri are piesa) și apasă butonul Wireframe din colțul din dreapta sus: peste model apare rețeaua de triunghiuri. Pe suprafețele plane, triunghiurile sunt mari și rare: o față dreaptă are nevoie doar de câteva. Pe rotunjimea găurii sunt mici și dese, deoarece un cerc nu poate fi redat din triunghiuri decât prin aproximare: cu cât sunt mai multe, cu atât conturul este mai rotund. Acolo se vede cel mai bine de ce STL-ul este o rețea de suprafețe, nu un desen. Brelocul nostru a rezultat cu 6.374 de triunghiuri, un număr nesemnificativ pentru un calculator, dar care descrie exact forma ta.

STL în Wireframe: suprafața devine o plasă de triunghiuri



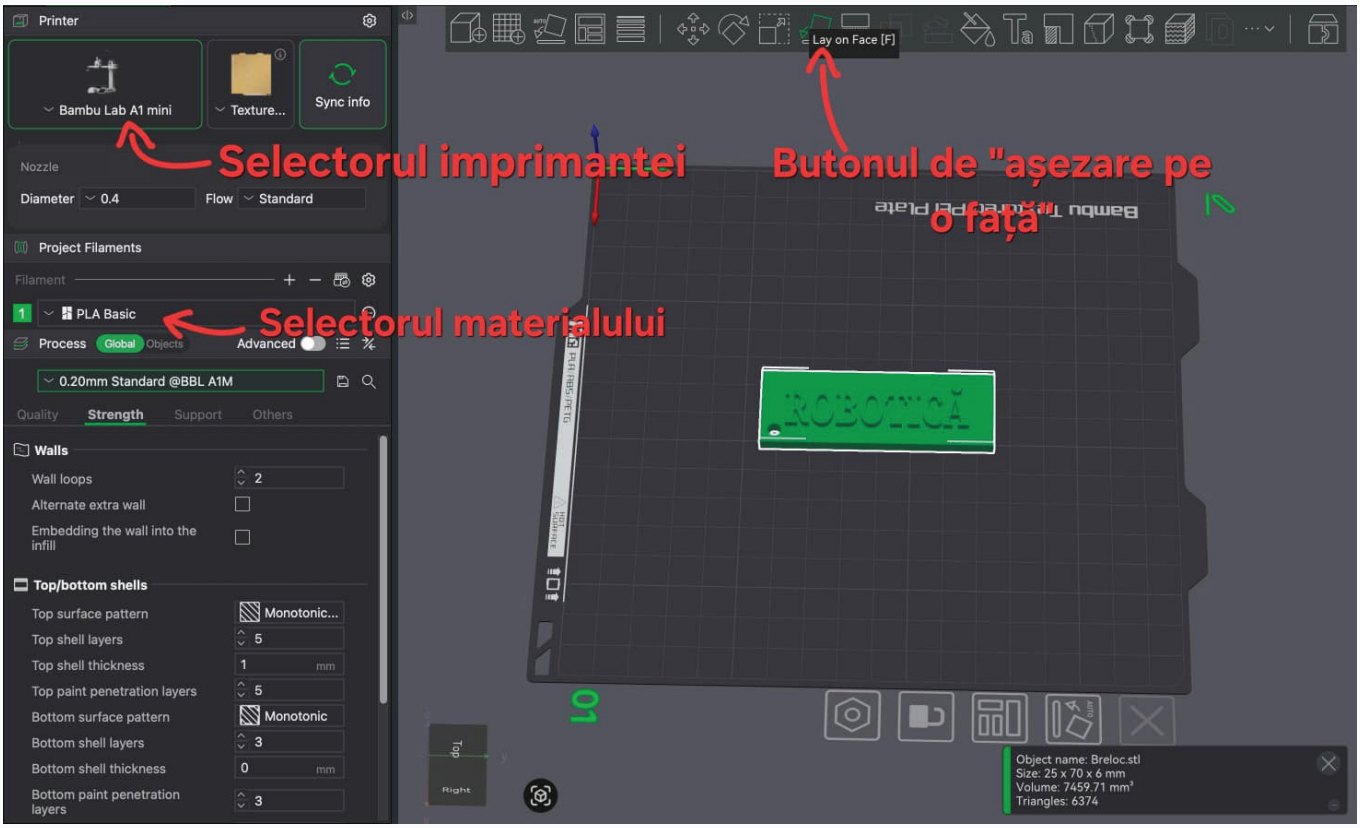
Vizualizatorul are și o casetă în care poți cere unui sistem de inteligență artificială să îți modifice modelul. Nu o folosim: modelul se modifică acolo unde a fost construit, în Tinkercad.

- [ ] Am fișierul STL al brelocului

2 · Feliază-l în Bambu Studio (demonstrație la ecran)

Pașii, în ordine: 1) se deschide Bambu Studio și se alege imprimanta din lista aflată în stânga sus (de exemplu, Bambu Lab A1 Mini); 2) fișierul STL se trage cu mouse-ul peste masa virtuală, iar piesa apare pe ea; dacă nu stă drept, butonul de așezare pe față o poziționează corect; 3) se alege materialul: PLA; 4) se verifică cele trei setări din lecție: înălțimea stratului 0,2 mm · umplerea 15 % · suporturile oprite (brelocul nu are nicio porțiune suspendată în aer); 5) se apasă Slice.

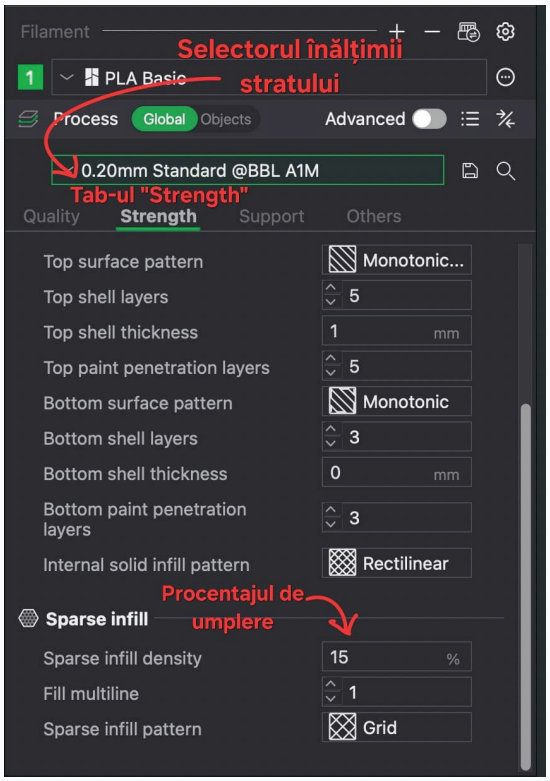
Bambu Studio: imprimanta, materialul și orientarea piesei



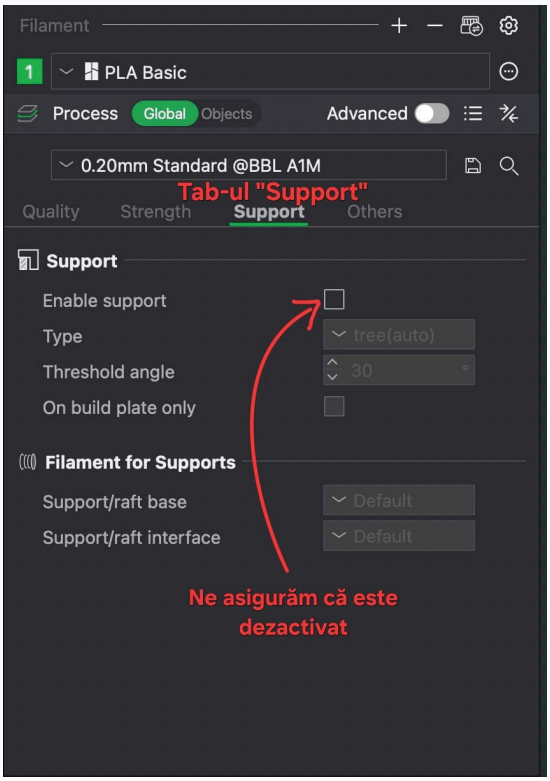
Rezultatul este o previzualizare pe care o poți derula strat cu strat, ca la un tomograf, plus cele două valori care contează în orice atelier: durata estimată și cantitatea de filament, în grame. Butonul de export scrie G-code-ul pe cardul imprimantei sau îl trimite direct către ea.

Cele trei setări în Bambu Studio

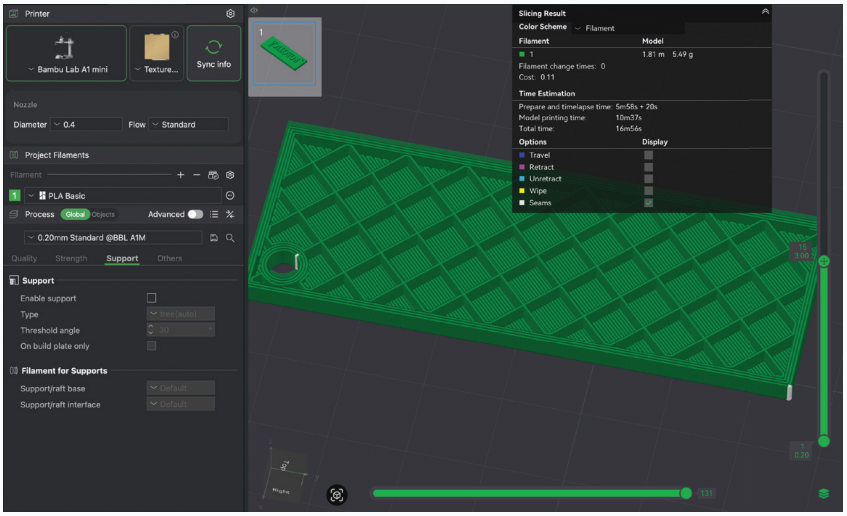
Strat + umplere



Suporturi oprite



Rezultat după “Slice”, la stratul 15 din 30



- [ ] Am văzut piesa feliată, durata și gramajul

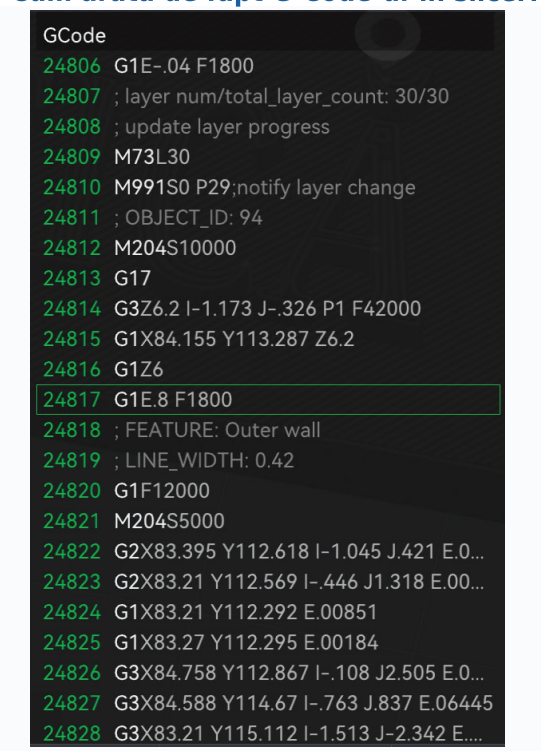
VITRINA · G-CODE · CUM ARATĂ, DE FAPT, PROGRAMUL IMPRIMANTEI

VITRINA · G-CODE

```
; layer num/total_layer_count: 30/30    ; slicerul își notează: începe stratul 30 din 30
G1 Z6 F1200                             ; ridică duza la 6 mm, înălțimea ultimului strat
G1 X84.155 Y113.287 F12000              ; deplasare la punctul (84,155 , 113,287) de pe masă
G1 X83.21 Y112.292 E0.00851             ; deplasare până în punctul următor, extrudând plastic
; FEATURE: Outer wall                   ; ce desenează acum: peretele exterior al piesei
```

Trei litere fac toată treaba: X și Y indică poziția pe masă; Z, înălțimea duzei; E, cantitatea de filament extrudată pe acel traseu; F, viteza de deplasare. Rândurile care încep cu ; sunt comentarii, exact ca # în Python. Instrucțiunile G2 și G3, reprezintă aceleași deplasări, dar pe arc de cerc, pentru rotunjimi. Un breloc are câteva zeci de mii de rânduri de acest fel.

Cum arată de fapt G-Code-ul în slicer:



TINKERCAD 3D · ATELIERUL

Aici se naște forma și tot de aici pleacă STL-ul: din contul tău, gratuit, de pe orice calculator cu internet.

SLICERUL · TRADUCĂTORUL

Bambu Studio (sau oricare altul) primește STL-ul, îl taie în straturi și scrie G-code-ul. Îl folosești doar atunci când există o imprimantă la capătul drumului.

Concluzia: lanțul complet al obiectului: idee → model (Tinkercad) → STL → slicer → G-code → piesă în palmă. În Modulul 5, piesele robotului vostru parcurg exact acest drum.

SECȚIUNEA 03 · DEPANARE

⚠ Exportul sau felierea nu merge?

1) Butonul Export nu oferă .STL → ești în Circuits, nu în 3D Designs; verifică în ce atelier lucrezi. 2) Fișierul exportat este gol sau conține doar o parte → aveai o formă selectată; execută clic pe fundal pentru a deselecta tot și alege „Everything in the design”. 3) Slicerul semnalează erori la model → piesele tale doar se ating, fără să se suprapună; revino în Tinkercad, suprapune-le cu 0,2-0,5 mm și grupează din nou. 4) Piesa apare uriașă sau minusculă în slicer → verifică unitatea de import: STL-ul din Tinkercad este în milimetri. 5) Durata estimată este de multe ore pentru un breloc → înălțimea stratului sau umplerea sunt mult peste valorile lecției; revino la 0,2 mm și 15 %.

SECȚIUNEA 04 · JOCULEȚUL LECȚIEI

Cum se joacă (8 minute, pe echipe)

Fiecare echipă primește o comandă de „atelier de tipărit” și decide cele trei setări, împreună cu argumentul: o figurină de șah cu detalii fine (strat? umplere?) · un braț de barieră care va fi solicitat zilnic (umplere?) · un prototip rapid, pentru verificarea dimensiunilor, gata până la finalul orei (strat? umplere minimă?) · o boltă de pod în miniatură (are nevoie de suporturi sau poate fi reproiectată?). Echipele își prezintă „bonul de comandă” completat, clasa contestă cu argumente, iar profesorul joacă rolul clientului nemulțumit. Concluzia serioasă a jocului: setările sunt compromisuri între timp, material, rezistență și finețe, iar fiecare piesă își are echilibrul ei.

Varianta Kahoot: situații cu setări de ales, cu 4 variante fiecare; linkul și codul QR se află în folderul lecției
Varianta pe foaie: fișa 4.6-J: trei „bonuri de comandă” de completat; strat · umplere · suporturi · argumentul
Varianta fără ecrane: chiar jocul principal; bonuri tipărite din folderul lecției sau dictate

Provocare: calculele brelocului

Nivel 1. Brelocul tău are 6 mm în punctul cel mai înalt (bază + relief). Câte straturi are la 0,2 mm? Dar la 0,1 mm?

Nivel 2. Piesa din provocarea Lecției 4.5 (suportul de senzor sau brațul de barieră): completează-i singur bonul de comandă, cu cele trei setări și argumentul, apoi păstrează-l. Este prima pagină din documentația proiectului vostru din Modulul 5.

3 întrebări rapide

1. Ce descrie un fișier STL și din ce este alcătuită „rețeaua” lui?
2. Ce primește și ce produce un slicer?
3. Când are nevoie o piesă de suporturi și care este alternativa mai elegantă?

Răspunsurile sunt la finalul modulului.

DICȚIONAR · CUVINTELE NOI DE AZI

STL: formatul universal: suprafața modelului, sub formă de rețea de triunghiuri

slicer: programul care taie modelul în straturi și scrie instrucțiunile imprimantei

Bambu Studio: slicerul gratuit folosit în manual pentru demonstrații

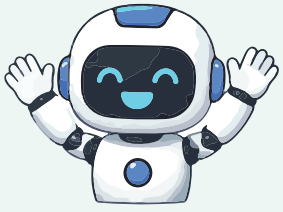
G-code: programul imprimantei: listă de instrucțiuni executate în ordine

umplere (infill): structura de tip fagure din interiorul piesei; tipic 15 %

suporturi: schelele tipărite sub porțiunile care ar atârna în aer

Ce trebuie să știi după această lecție

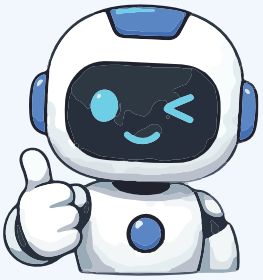
- ✓ STL = suprafața modelului în triunghiuri; se exportă din Tinkercad în trei clicuri: Export → ce exporți → .STL
- ✓ Slicerul face STL → straturi → G-code, după setările tale
- ✓ Cele trei setări-cheie: înălțimea stratului · umplerea · suporturile; toate sunt compromisuri
- ✓ Slicingul este demonstrativ: îl parcurgi doar dacă ai o imprimantă; STL-ul îl poate obține oricine



FINAL DE MODUL · MODULUL 4

Ai terminat Modulul 4!

Acum șase ore aveai componente separate. Astăzi ai dispozitive complete (bariera care vede, ventilatorul care simte căldura, lampa care ascultă și de senzori, și de tine), plus o competență nouă: poți proiecta obiecte reale, milimetru cu milimetru, și cunoști tot drumul lor până în palma ta. A mai rămas un singur pas, cel mai frumos: în Modulul 5, toate acestea se unesc într-un proiect de echipă: robotul vostru, documentația lui, prezentarea lui. De data aceasta nu mai construiești o lecție. Construiești ceva al vostru.



Lanțul complet al obiectului

1 IDEE → 2 MODEL TINKERCAD → 3 STL → 4 SLICER → 5 G-CODE → 6 PIESĂ

De reținut: idee → model (Tinkercad) → STL → slicer → G-code → piesă în palmă. În Modulul 5, piesele robotului vostru parcurg exact acest drum.

În Modulul 5 nu mai construiești o lecție. Construiești ceva al vostru.

Răspunsuri la „3 întrebări rapide” · 4.1-4.3

Lecția 4.1

1. INTRAREA (ce senzor?), DECIZIA (ce condiție?), IEȘIREA (ce actuator?).
2. Este planul scris în limba română: dacă planul este limpede, codul devine o simplă traducere; greșelile de gândire se descoperă pe hârtie, nu în fire.
3. PIR-ul răspunde doar „este mișcare / nu este”, deci bariera reacționează la orice trecere; ultrasonicul măsoară distanța, deci bariera urcă doar când ținta este suficient de aproape.

Lecția 4.2

1. Adaugă ramuri intermediare între „dacă” și „altfel”; condițiile se verifică de sus în jos și se execută prima adevărată.
2. Deoarece prima ramură adevărată câștigă: cu pragul mic scris primul, condiția „grade > 26” ar fi adevărată și la 35 °C, iar treapta maximă nu ar mai fi atinsă niciodată.
3. Sub aproximativ 15 % din scara de comandă, motorul nu învinge inerția și frecarea (Lecția 2.5), deci treapta de mijloc trebuie să fie de cel puțin 40-50 %.

Lecția 4.3

1. Când cel puțin una dintre condițiile legate este adevărată.
2. ȘI se evaluează primul, ca înmulțirea înaintea adunării: perechea (întuneric ȘI mișcare) se evaluează oricum împreună, înainte de SAU. Parantezele nu schimbă deci rezultatul aici, dar arată intenția dintr-o privire, iar la condiții mai lungi devin singura plasă de siguranță.
3. Calea prin care omul preia oricând controlul (butonul nostru); orice automatizare serioasă o are, de la lift la frâna de mână.

Răspunsuri la „3 întrebări rapide” · 4.4-4.6

Lecția 4.4

1. Aditivă = adaugă material doar unde este necesar, strat cu strat; prin îndepărtare = pleacă de la un bloc și taie ce este în plus.
2. Deoarece orice formă complicată devine simplă atunci când este tăiată în felii: fiecare strat este doar un desen plan, ușor de executat cu două motoare.
3. Pașii lor ficși (1,8°), transformați prin curele și șuruburi în deplasări de sutimi de milimetru, dau precizia fiecărui strat.

Lecția 4.5

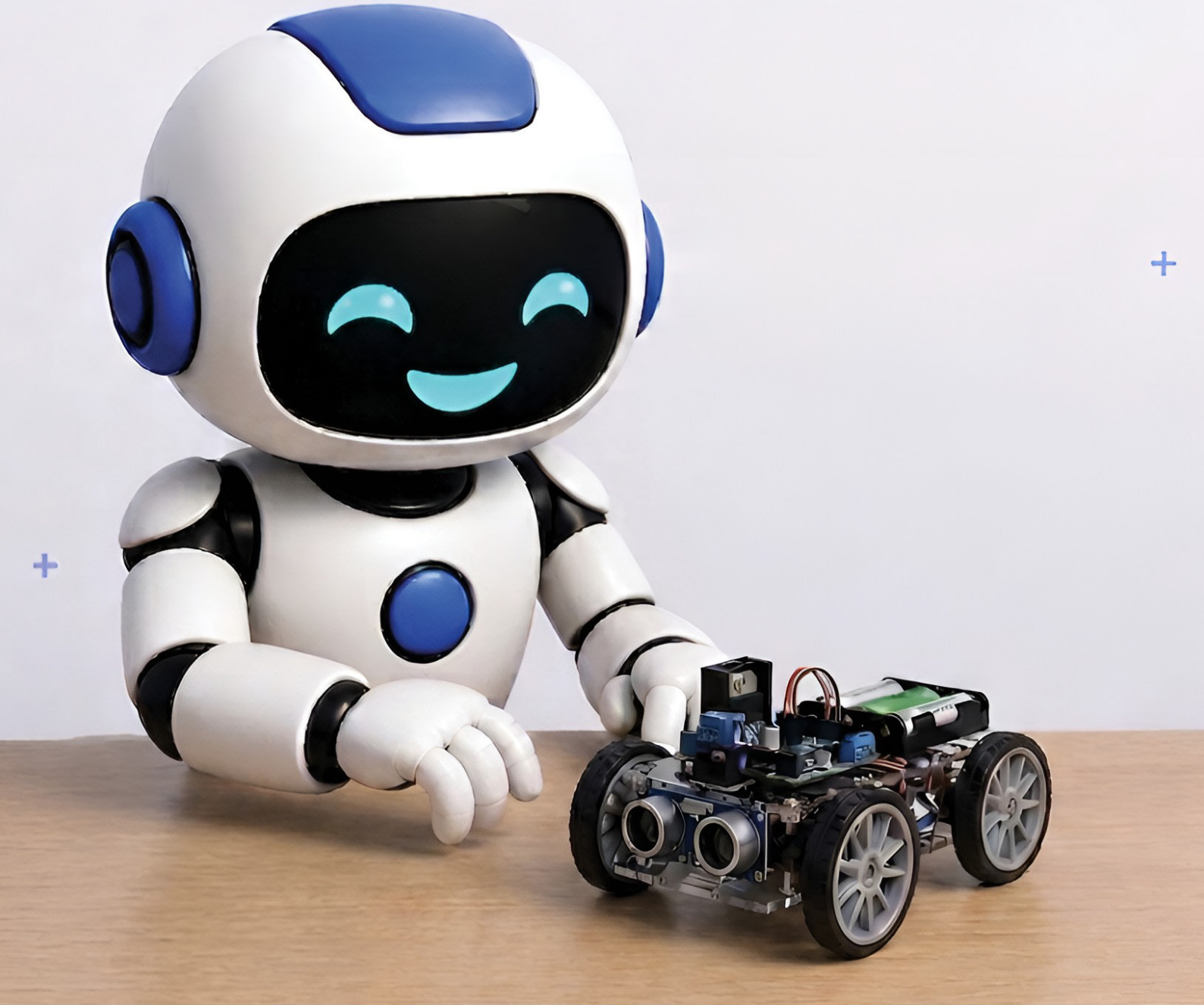
1. Aduci și potrivești forme (cu dimensiuni exacte) · scobești cu găuri (Hole) · grupezi (Group).
2. Acolo unde gaura se suprapune cu solidele, materialul dispare, iar scobitura devine parte din obiectul unic.
3. Pentru că proiectezi la scară reală: piesa trebuie să se potrivească milimetric cu senzori, șuruburi, cornuri de servomotor.

Lecția 4.6

1. Suprafața modelului, acoperită cu o plasă de mii de triunghiuri minuscule.
2. Primește STL-ul și setările; produce piesa feliată și G-code-ul, adică programul imprimantei, împreună cu durata și consumul estimate.
3. Când are porțiuni suspendate, înclinate cu peste aproximativ 45° față de verticală; alternativa elegantă este reproiectarea piesei, astfel încât să nu aibă nicio porțiune în aer.

Modulul 5

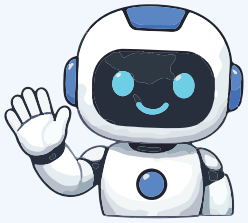
PROIECTUL FINAL



MODULUL 5 · PROIECTUL FINAL

Modulul 5. Proiectul final

Patru module, un singur scop: acesta. De aici încolo nu mai construiești lecții, ci construiești, împreună cu echipa ta, un proiect al vostru: cu senzori, decizii, actuatori, o piesă proiectată 3D, o documentație tehnică și o prezentare. Nu mai este nimic nou de învățat. Tot ce urmează folosește strict ce cunoașteți deja din Modulele 1-4; dificultatea nu mai stă în cunoștințe, ci în a le face să lucreze împreună, până la capăt, fără ca cineva să vă spună la fiecare pas ce urmează.



MANUAL DE ROBOTICĂ · CLASELE IX-XII · TINKERCAD

Cum funcționează acest modul

Nu mai există lecții. Există un proiect ales de voi, un parcurs de lucru în patru pași și două produse care se predau odată cu el: documentația tehnică și prezentarea. Profesorul distribuie pașii pe câte săptămâni are nevoie clasa; manualul stabilește structura, nu ritmul.

Regulile de bază: proiectul folosește doar noțiuni din Modulele 1-4 · totul se simulează integral în Tinkercad · tema o alege echipa · grila de punctaj este publică de la început și o găsiți spre finalul modulului, ca să știți din prima zi după ce criterii veți fi evaluați.

SECȚIUNEA 01 · PROIECTUL VOSTRU

Ce aveți de construit

Un sistem care simte, decide și acționează: lanțul complet pe care l-ați construit element cu element în modulele anterioare, reunit acum într-un obiect cu o funcție clară. Tema o alegeți voi. Nu există un proiect „corect”: există proiecte care trec testul de mai jos și proiecte care nu îl trec, iar diferența se vede din prima zi, nu în ultima.

Testul celor cinci criterii · pentru orice idee de proiect

1 · Are cel puțin un senzor, o decizie și un actuator, adică lanțul complet.

2 · Se simulează integral în Tinkercad, fără altă platformă.

3 · Folosește doar noțiuni din Modulele 1-4.

4 · Are cel puțin o piesă proiectată 3D de echipă (carcasă, suport, braț).

5 · Prima versiune funcțională este gata până la Pasul 3; ambiția rămâne pentru îmbunătățiri, nu pentru structura de bază.

O idee care trece toate cele cinci criterii este un proiect bun, indiferent cui îi aparține.

Piesa 3D se realizează separat de circuit

În Tinkercad, circuitele și modelele 3D se construiesc în două spații de lucru diferite, care nu se pot uni: nu veți putea monta piesa peste montajul din simulare, iar simularea nu va detecta niciodată existența piesei. Este o limitare a instrumentului, nu o omisiune a voastră.

Piesa rămâne, așadar, un model: o dovadă vizuală, nu un obiect care se montează efectiv pe circuit. Acest lucru nu o face decorativă; ea este o componentă deplină a proiectului, gândită pentru montajul vostru anume: carcasa care protejează placa, suportul care menține senzorul la unghiul potrivit, brațul care se prinde pe cornul servomotorului.

Se predă separat: capturi ale modelului, dimensiunile, motivul formei alese și felul în care s-ar monta peste componentele reale. Are capitolul ei în documentație, slide-ul ei în prezentare și punctajul ei în grilă. Circuitul dovedește că sistemul ia decizii; piesa 3D dovedește că v-ați gândit și la felul în care el funcționează în lumea reală.

De unde porniți

Lista de mai jos nu este un meniu din care trebuie să alegeți, ci o serie de puncte de plecare. Descrierile precizează ce face proiectul și de ce fel de componente are nevoie, nu care sunt exact acelea: alegerea componentei potrivite face parte din proiect, iar toate v-au trecut deja prin mână în Modulele 1-4. Cea mai bună idee rămâne, oricum, cea care vă aparține: o problemă pe care o observați zilnic și pe care nimeni nu s-a apucat să o rezolve.

PROIECTUL	CE FACE	DE CE FEL DE COMPONENTE ARE NEVOIE	NIVEL
Bariera de parcare	Se ridică singură când se apropie un autovehicul și coboară după ce acesta a trecut, cu semnalizare luminoasă și sonoră.	un senzor care măsoară distanța · un actuator care se oprește la un unghi precis · ieșiri luminoase și sonore	●●○
Sera inteligentă	Udă atunci când solul este uscat și ventilează atunci când este prea cald: două decizii independente, în același sistem.	doi senzori care citesc mediul (umiditatea solului, temperatura) · două ieșiri comandate separat	●●●
Semaforul cu buton de pieton	Funcționează pe ciclul lui normal, dar la apăsarea butonului acordă verde pietonului, cu semnal de traversare.	o intrare digitală · mai multe ieșiri luminoase · o sursă de sunet · lucrul cu timpi	●●○
Alarma de cameră	Se armează dintr-un buton și emite semnal sonor doar dacă detectează mișcare pe întuneric: două condiții, o singură decizie.	un senzor care detectează mișcarea · un senzor care măsoară lumina · o intrare de armare · o ieșire sonoră	●●○
Ventilatorul cu carcasă proprie	Suflă cu atât mai tare cu cât e mai cald și se oprește singur când temperatura scade.	un senzor de temperatură · un actuator care se rotește continuu, comandat în trepte · o piesă 3D desenată de voi	●●○
Ușa automată de magazin	Se deschide când clientul este aproape și se închide după câteva secunde, nu instantaneu.	un senzor de proximitate · un actuator care se rotește controlat · o temporizare gândită	●●○

PROIECTUL	CE FACE	DE CE FEL DE COMPONENTE ARE NEVOIE	NIVEL
Lampa de veghe adaptivă	Cu cât camera este mai întunecată, cu atât luminează mai puternic, iar comanda poate fi trecută oricând pe manual.	un senzor de lumină · o ieșire cu intensitate reglabilă continuu · o intrare de comandă	●○○
Semnalizarea pentru bicicletă	Două butoane pe ghidon, două seturi de lumini care clipesc independent, plus carcasa care le fixează pe cadru.	două intrări digitale · ieșiri luminoase intermitente · o piesă 3D funcțională	●○○
Stația meteo de fereastră	Măsoară temperatura și lumina, le arată permanent și avertizează la praguri de caniculă sau îngheț.	doi senzori de mediu · o formă de afișare a valorilor · ieșiri de avertizare	●●●
Iluminatul stradal inteligent	Ziua stă stins, noaptea stă în semi-intensitate și se aprinde complet doar când trece cineva.	un senzor de lumină · un senzor care detectează mișcarea · o ieșire cu intensitate reglabilă	●●○
Numărătorul de la intrare	Numără câte persoane au trecut prin dreptul lui și avertizează când s-a atins limita de ocupare.	un senzor care sesizează trecerea · o formă de afișare a numărului · o ieșire de avertizare	●●○
Seiful cu cod din butoane	Se deschide doar dacă butoanele sunt apăsate în ordinea corectă; la greșeală se resetează și avertizează.	mai multe intrări digitale · memorarea unei secvențe · un actuator care blochează sau eliberează · o ieșire de avertizare	●●●

Echipa și rolurile

Orice echipă de ingineri împarte munca pe roluri, iar ale voastre, pentru 3-5 membri, sunt cinci: coordonatorul (urmărește pașii, jurnalul și buna colaborare), electronistul (circuitul și cablarea), programatorul (pseudocodul și codul), proiectantul 3D (piesa tipăribilă) și responsabilul cu documentația și prezentarea. În echipele mici, o persoană poate îndeplini două roluri; este normal. Două reguli fac diferența: fiecare rol are un nume de persoană alături și toți lucrează la toate, deoarece rolul indică cine răspunde, nu cine are voie să se implice.

- coordonatorul
- electronistul
- programatorul
- proiectantul 3D
- documentația + prezentarea

SECȚIUNEA 02 · PARCURSUL DE LUCRU

Patru pași, în ordinea asta

Ordinea nu este întâmplătoare. Fiecare pas se încheie verificat, pentru ca următorul să pornească de pe o bază sigură, astfel încât, atunci când ceva nu funcționează (și se va întâmpla, face parte din meserie), să știți exact unde să căutați. Un pas nu este „gata” pentru că ați lucrat la el, ci pentru că bifa de la finalul lui poate fi pusă în mod onest.

PATRU PAȘI · ÎN ORDINEA ASTA

1 FIȘA

2 SIMTE + DECIDE

3 ACTUATORI

4 TEST + 3D

1 · Fișa de proiect și rolurile

O pagină scrisă înainte de primul fir, cu șapte rubrici: numele proiectului și al echipei · problema pe care o rezolvă, într-o singură propoziție · schema-bloc (intrare → decizie → ieșire) · lista componentelor · placa aleasă și argumentul alegerii · piesa 3D propusă · rolurile, fiecare cu un nume alături. Se semnează de toată echipa și devine, nemodificată, capitolul 2 al documentației.

- ☐ Fișa este completă, semnată, iar proiectul trece toate cele cinci criterii

2 · Partea care simte și decide

Se construiește prima, separat, fără actuatori. Cablați senzorii, măsurați valorile reale în situațiile care contează (aproape/departe, lumină/întuneric, cald/rece) și stabiliți pragurile din aceste măsurători, nu prin estimare. Decizia se scrie mai întâi în pseudocod, apoi în cod, iar în locul actuatorilor conectați un LED martor: dacă martorul se aprinde exact când trebuie, partea de decizie funcționează corect.

- ☐ Senzorii citesc, pragurile sunt măsurate și notate, LED-ul martor răspunde corect

Lanțul complet pe care îl verifică Pasul 2

1 **SENZOR** — SIMTE

2 **DECIZIE** — ALEGE

3 **ACTUATOR** — ACȚIONEAZĂ

3 · Actuatorii și integrarea

Actuatorii intră pe rând, nu toți odată: fiecare se testează izolat, apoi ia locul LED-ului martor. Nu uitați regulile care nu iartă: motorul se comandă prin tranzistor și primește dioda lui de protecție (2.3), firul de alimentare al servomotorului se conectează la pinul de alimentare al plăcii, niciodată la un pin de comandă (2.6), iar motorul primește sursă proprie, cu masă comună, atunci când pe aceeași placă lucrează și un senzor care măsoară (4.2). Adăugarea câte unui actuator o dată înseamnă că orice defecțiune nouă are un singur suspect posibil.

- ☐ Toți actuatorii răspund deciziei, în cadrul sistemului complet

4 · Testarea, piesa 3D și înghețarea

„Funcționează atunci când îl prezint eu” nu înseamnă testare. Scrieți lista de teste înainte: toate combinațiile normale de intrări, plus cazurile-limită, adică exact pe prag, în prima secundă după pornire, cu senzorul acoperit, cu două condiții schimbate simultan. Ce nu trece testul se repară sau se consemnează onest. La final, piesa 3D se modelează și se exportă, în spațiul 3D, separat de circuit; versiunea se salvează și urmează decizia matură: înghețarea proiectului. De aici încolo se repară, nu se adaugă; ideile noi merg la capitolul „versiunea 2”.

- ☐ Lista de teste este parcursă integral, piesa 3D este exportată, proiectul este înghețat

Jurnalul de proiect · trei rânduri după fiecare ședință

Ce am făcut · ce nu a funcționat · ce urmează. Atât. Pare un detaliu mărunt, dar este singurul element din tot modulul care lucrează pentru voi și după ce ați închis calculatorul: din el se scriu aproape neschimbate capitolul de testare și istoricul din documentație, tot din el rezultă răspunsurile la întrebările juriului. Echipele care nu îl țin rescriu totul la final, din memorie, iar acest lucru se observă.

⚠ Cele două capcane care se repetă în fiecare an

Prima: „hai să adăugăm și motorul, că tot funcționează…”, adică funcții adăugate peste o parte care încă nu este verificată. Când apoi ceva nu funcționează, nu mai puteți ști dacă defecțiunea vine de la componenta nouă sau de la ceva mai vechi.

A doua: documentația scrisă în ultima seară. Se recunoaște imediat, după capturile lipsă, timpurile verbale amestecate și stilul unei singure persoane, scris în grabă. Soluția este aceeași pentru ambele capcane: pas cu pas, cu jurnalul la zi.

SECȚIUNEA 03 · DOCUMENTAȚIA TEHNICĂ

Ce este o documentație tehnică

Documentația tehnică este descrierea completă și ordonată a unui produs: ce face, cum este construit, cum funcționează, cum a fost verificat și ce limite are. În inginerie nu este o sarcină suplimentară, adăugată la final, ci face parte din produs. Un dispozitiv fără documentație nu poate fi reparat, îmbunătățit sau preluat de altcineva; el rămâne inutilizabil odată cu plecarea autorilor lui.

Testul unei documentații bune este simplu și exigent: un cititor care nu a văzut niciodată proiectul vostru trebuie să îl poată reconstrui doar din ea, fără prezența voastră. Trei reguli o ajută să treacă testul. Capturile sunt mai eficiente decât paragrafele: o schemă adnotată, cu săgeți și etichete, transmite cât o pagină de text. Se scrie pe parcurs, din jurnal, nu în ultima noapte. Iar onestitatea se punctează: „nu a funcționat, am înțeles de ce, am reparat astfel” este cea mai convingătoare formulare pe care o poate citi un inginer.

CAPITOLUL	CE CONȚINE
1 · Coperta	Numele proiectului, numele echipei și ale membrilor, școala și anul, plus o imagine mare a sistemului vostru.
2 · Fișa de proiect	Chiar pagina semnată la primul pas: problema, soluția pe scurt, placa aleasă și argumentul alegerii.
3 · Schema-bloc și piesele	Desenul intrare → decizie → ieșire, plus lista completă a componentelor, fiecare cu rolul ei într-o propoziție.
4 · Schema circuitului	Captura din simulator, adnotată: ce se conectează, unde și de ce. Aici se răspunde la întrebarea „cum îl reconstruiesc?”.
5 · Programul, explicat	Codul complet, plus 5-8 comentarii-cheie asupra deciziilor importante: pragurile, condițiile compuse, temporizările.

CAPITOLUL	CE CONȚINE
6 · Piesa 3D	Capturile modelului, dimensiunile, motivul formei alese și felul în care s-ar monta peste componentele reale.
7 · Testarea	Lista de teste parcursă, cu rezultate, inclusiv cazurile-limită identificate, reparate sau asumate explicit.
8 · Istoricul proiectului	Cinci sau șase însemnări reprezentative din jurnal, în ordine cronologică, inclusiv eșecurile și modul în care le-ați depășit.
9 · Roluri și contribuții	Cine ce a făcut, concret și onest. Juriul întreabă mereu, iar răspunsurile evazive se aud.
10 · Concluzii și versiunea 2	Ce ați învățat, ce limite are sistemul și ce ar cuprinde o versiune următoare, inclusiv ideile amânate pe parcurs.

> ⚠ **Regula împotriva amânării**
> Din prima zi de redactare, fiecare capitol primește un responsabil și un termen, notate pe el. Capitolele 3-6 revin celor care le-au construit (electronistul, programatorul, proiectantul), 7 și 8 se scriu din jurnal, iar 1 și 10 se redactează la final, împreună. Documentația nu se improvizează la sfârșit, ci se construiește treptat.

SECȚIUNEA 04 · PREZENTAREA PROIECTULUI

Cum se prezintă un proiect

Documentația convinge un cititor; prezentarea convinge o sală. Este aceeași poveste spusă în două forme: una se citește în liniște și poate fi recitită oricând, cealaltă se ascultă o singură dată, fără posibilitatea de a reveni. De aceea, prezentarea nu este documentația proiectată pe perete, ci varianta ei cu multe imagini și puțin text.

O prezentare de proiect are aceeași structură, indiferent de temă: pornește de la problemă, arată soluția în ansamblu, apoi urmează lanțul complet (cum simte, cum decide, cum acționează) și se încheie cu ce nu a funcționat și ce urmează. Schema de mai jos este exact această structură, împărțită pe slide-uri.

SLIDE	CE CONȚINE (ȘI CE NU)
1 · Titlul	Numele proiectului, echipa și o imagine mare a sistemului. Fără fraze.
2 · Problema	O imagine din viața reală și o singură propoziție: ce doare.
3 · Soluția	Schema-bloc, mare și clară. O explicați voi; slide-ul doar o afișează.
4 · Cum simte	Capturi din simulare cu valorile senzorilor vizibile și pragurile alese.
5 · Cum decide	Condiția-cheie scrisă cu litere mari, în pseudocod, nu tot codul. De unde provin pragurile.
6 · Cum acționează	Actuatorii în acțiune: o captură sau, ideal, înregistrarea video demonstrativă.
7 · Piesa 3D	Modelul prezentat din mai multe unghiuri și motivul formei alese: ce problemă rezolvă piesa.
8 · Ce nu a mers	Slide-ul care câștigă respectul juriului: o defecțiune memorabilă și modul în care ați rezolvat-o.
9 · Versiunea 2	Ideile amânate pe parcurs: dovada că gândirea nu s-a oprit la termenul de predare.
10 · Mulțumim + demo	O frază de final memorată și trecerea la demonstrația live.

Regulile care despart prezentările bune de slide-urile încărcate cu text

O idee pe slide · maximum 20 de cuvinte pe slide · imaginile sunt ale proiectului vostru, nu preluate de pe internet · nimeni nu citește de pe ecran, deoarece slide-ul este suportul vizual, iar voi sunteți prezentarea · fiecare membru vorbește cel puțin o dată · primele zece secunde se memorează, restul se povestește · privirea în sală, nu în ecran · totul încape în 5 minute, cronometrate la repetiție, nu estimate aproximativ.

O IDEE PE SLIDE · MAXIMUM 20 DE CUVINTE · 5 MINUTE CRONOMETRATE

⚠ Regula demonstrațiilor live și planul B

Orice demonstrație live poate eșua exact pe scenă: conexiunea la internet cade, simularea se blochează, emoțiile își spun cuvântul. De aceea, regulă fără excepție: înregistrați din timp o secvență video de 30 de secunde cu sistemul funcționând în simulare și salvați-o în două locuri. Dacă demonstrația live reușește, înregistrarea rămâne nefolosită. Dacă nu, o porniți, iar prezentarea continuă; juriul apreciază calmul, nu defecțiunea.

SECȚIUNEA 05 · ZIUA DEMO ȘI EVALUAREA

Cum se desfășoară și după ce se punctează

Ordinea echipelor se stabilește prin tragere la sorți. Fiecare echipă are 5 minute de prezentare și demonstrație, urmate de 2 minute de întrebări. Juriul este format din profesor și doi invitați: un alt profesor, un părinte, un elev dintr-o clasă mai mare. Grila de mai jos este publică din prima zi a modulului: nimic din ceea ce se punctează nu vă ia prin surprindere.

CRITERIUL	PUNCTE	CE URMĂREȘTE JURIUL
Funcționarea sistemului (demonstrația)	30	lanțul complet funcționează; cazurile-limită nu îl blochează
Documentația tehnică	25	completă, clară, onestă; proiectul poate fi reconstruit fără prezența voastră
Prezentarea	20	structura, imaginile proprii, vorbitul liber, toți membrii
Ingeniozitatea și piesa 3D	15	soluții proprii, piesă gândită pentru montajul vostru și argumentată
Lucrul în echipă	10	roluri reale, jurnal ținut, contribuții oneste
TOTAL	100	·

Cele cinci întrebări clasice ale oricărui juriu · pregătiți-le de acasă

1 · De ce ați ales acest senzor și nu altul? 2 · Ce se întâmplă dacă… (un caz-limită)? 3 · Care a fost cel mai greu moment și cum l-ați trecut? 4 · Cine ce a făcut, concret? 5 · Ce ați schimba dacă ați lua-o de la capăt?

Nu sunt capcane și nu pot fi memorate: echipa care a ținut jurnalul și a redactat documentația în mod onest are deja toate răspunsurile formulate.

DICȚIONARUL MODULULUI

documentație tehnică: descrierea completă a produsului: ce face, cum este construit, cum a fost verificat

fișa de proiect: pagina care definește proiectul: problema, schema, componentele, rolurile

schemă-bloc: desenul intrare → decizie → ieșire, fără fire și fără cod

piesa 3D: modelul desenat separat de circuit: se prezintă și se argumentează, nu se montează în simulare

jurnal de proiect: trei rânduri după fiecare ședință: realizat · nu a funcționat · urmează

LED martor: LED-ul care ține locul actuatorilor cât timp se verifică decizia

caz-limită: situația de la marginea firescului, exact acolo unde se ascund defectele

înghețarea proiectului: momentul după care se repară, dar nu se mai adaugă nimic

captură adnotată: imagine cu săgeți și etichete; transmite cât o pagină de text

plan B: înregistrarea video de 30 de secunde cu simularea funcțională, pentru situația în care demonstrația live eșuează

pitch: prezentarea scurtă și convingătoare a unei idei, în sub un minut

Lista de autoverificare · de parcurs înainte de predare

Nicio bifă pusă de complezență. Un rând nebifat cu o zi înainte se poate remedia; unul nebifat pe scenă, nu.

- ☐ Proiectul trece toate cele cinci criterii, iar fișa este semnată de toată echipa
- ☐ Lanțul complet funcționează în simulare: senzorii citesc, decizia decide, actuatorii răspund
- ☐ Pragurile din cod provin din măsurători consemnate, nu din estimări
- ☐ Lista de teste este scrisă și parcursă integral, cu cazurile-limită incluse
- ☐ Piesa 3D este modelată de echipă și exportată, cu forma și dimensiunile argumentate
- ☐ Versiunea finală este salvată separat, iar proiectul este înghețat
- ☐ Documentația are toate cele 10 capitole, cu capturi adnotate și un istoric onest
- ☐ Un coleg din altă echipă a citit documentația și a înțeles proiectul fără explicații de la voi
- ☐ Prezentarea începe în 5 minute cronometrate, iar fiecare membru vorbește cel puțin o dată
- ☐ Înregistrarea video pentru planul B există și este salvată în două locuri

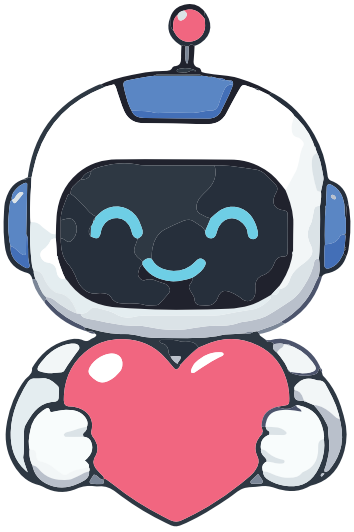
FINALUL MANUALULUI

Ai terminat manualul!

Cu un an în urmă, „robotică” era un cuvânt din filme. Astăzi știi ce determină curentul să circule prin fire și cum îl limitezi, ai pornit motoare și le-ai schimbat sensul, ai învățat o placă să perceapă mediul și să ia decizii, ai proiectat obiecte care pot fi ținute în palmă, iar la final ai dus, împreună cu echipa ta, un proiect întreg de la o idee scrisă pe o fișă până într-o sală care aplaudă. Nimic din toate acestea nu se uită. Păstrați tot: versiunile, jurnalul, documentația, prezentarea. Data viitoare când începeți un proiect, și va exista o dată viitoare, nu veți mai porni de la zero.

DRUMUL COMPLET · MODULELE 1–5

1 MODULUL 1 → 2 MODULUL 2 → 3 MODULUL 3 → 4 MODULUL 4 → 5 MODULUL 5



ÎNCHEIERE · DUPĂ ULTIMA LECȚIE

Cât de departe ai ajuns

La prima lecție, un circuit era un desen greu de citit. Acum poți construi un sistem care măsoară mediul, ia o decizie și acționează asupra lui, poți explica de ce funcționează și poți repara ce nu funcționează.

DE LA PRIMUL LED LA UN PROIECT COMPLET

Să spunem lucrurile exact: nu ai devenit inginer și nici acesta nu a fost scopul. Un an de lucru nu înlocuiește experiența unui laborator, iar simulatorul, oricât de fidel, rămâne un simulator. Ai însă ceva ce puțini oameni au: baza. Cunoști noțiunile fundamentale, ai văzut cum se leagă între ele și ai dus un proiect până la capăt. De aici înainte, orice placă nouă, orice senzor și orice limbaj se învață mult mai repede, pentru că știi ce să cauți. Iar dacă vei alege un drum complet diferit, tot rămâi cu ceva. Robotica a fost, în fond, doar pretextul pentru câteva deprinderi de gândire care nu îi aparțin.

Ce rămâne cu tine, dincolo de robotică

Întâi planul, apoi execuția. Schema-bloc și pseudocodul te-au învățat să gândești o soluție înainte de a o construi.

Măsoară, nu estima. Pragurile din lecțiile cu senzori au venit din măsurători, nu din păreri. Aceasta este diferența dintre o impresie și un argument.

Schimbă un singur lucru odată. Când adaugi câte un actuator pe rând, orice defecțiune are un singur suspect. Aceeași metodă rezolvă și probleme care nu au nicio legătură cu firele.

Ce nu a funcționat este informație. Depanarea te-a obișnuit să cauți cauza, în loc să te enervezi pe efect.

Explică-ți munca. Documentația și prezentarea te-au învățat că o soluție pe care nu o poți explica altcuiva este pe jumătate terminată.

Echipa are nevoie de roluri, nu de eroi. Proiectul final a fost, de fapt, un exercițiu de lucru împreună.

Unde îți folosesc mai departe

- ✓ **La școală: la fizică, matematică și informatică vei recunoaște, deja lucrate cu mâna ta, noțiuni pe care alții le întâlnesc doar pe hârtie.**
- ✓ **Acasă: știi să citești o fișă tehnică, înțelegi de ce un aparat cere o anumită tensiune și de ce la alimentare nu se improvizează.**
- ✓ **Mai departe: dacă alegi un domeniu tehnic, pornești cu limbajul deja învățat. Dacă nu, rămâi cu metoda.**

Păstrează tot ce ai construit: circuitele, jurnalul, documentația, prezentarea. Data viitoare când vei porni ceva de la zero, vei descoperi că nu mai pornești chiar de la zero.

SUCCES ÎN TOT CE VEI CONSTRUI DE ACUM ÎNAINTE