Természettudományos gondolkodás fejlesztése Microbit vezérlővel, elektronikai elemekkel

A bemutatóban környezetünk technikai, technológiai megoldásait modellezzük Microbit vezérlőkkel, külső szenzorokkal, MI kamerával, elektronikai elemekkel. Egy-egy miniprojektben bemutatjuk a modell leírását , a működtetéshez készített programkódot, a modell funkcionális környezetét. A programozáshoz két programozási környezetet használunk. Az egyik a microbit.org online blokkprogramozási környezet, a másik a MicroPython MU editorral.

A mini projektek adaptációját a 12-16 éves korosztálynak ajánljuk szaktábor, tehetséggondozó program, projektmunka, témanap, témahét keretében. A tevékenységek több tantárgy tartalmait integrálják: természettudományok, technika és tervezés, vizuális kultúra, digitális kultúra.

Szabó János ereszx@gmail.com
Lévai Edit levaiedit65@gmail.com
Kasztiné Végh Tímea kasztitimi@gmail.com

Vulkáni energiák projekt

Magyar nyelvű videó:
https://www.youtube.com/watch?v=MLGC9rgsPNU
Angol nyelvű videó:
https://www.youtube.com/watch?v=_u0P1TytHQM

Modell bemutatása

A vulkánokban rejlő energiák hasznosítása napjainkra már realitássá vált, de a technológiai, technikai megoldások még nagy kihívást jelentenek. Egy vulkán kitörésekor óriási energia szabadul fel magas nyomású gőz, gázok és hő formájában, ami a földi életre káros hatással van. Egy nagyobb kitörés rövidtávú globális klímaváltozást is okozhat, aminek nyilvánvalóan negatív gazdasági és társadalmi hatásai is vannak. Ha ezt az energiát még a kitörések előtt ki tudnánk nyerni, az jelentős mennyiségű tiszta energia előállítására lenne alkalmas és ezzel környezetünket is óvnánk. Azonban a természet működésébe történő ilyen mértékű beavatkozás óriási felelősség, csakis kontrollált körülmények között lehetséges. A technikai, technológiai megvalósításához elengedhetetlen, hogy folyamatosan figyeljük, mérjük a vulkáni tevékenységeket, ami egyébként a kitörések előrejelzéséhez is hasznos információt nyújt.

Projektünkben ezt a mérési folyamatot modellezzük szimulált környezetben, programozott mikrovezérlőkkel, külső szenzorokkal. A mérési eredményeket egy távoli, 3D tervezett, földrengésbiztos vulkanológiai központban követjük nyomon grafikonokon.

Készítettünk egy vulkán modellt gipszből. A szenzorokat a vulkán modell környezetében helyezzük el, ez a mérőállomás. Itt fogjuk lehetőségeinkhez képest a vulkáni jelenségeket szimulálni és mérni.

A mérőállomáson hat szenzort helyezünk el egy Microbit vezérlő analóg lábaira csatlakoztatva: gáz, gőz, hő, tűz, rezgés, zajszint érzékelőket. A program feladata a paraméterek mérése és továbbítása Bluetooth kapcsolaton keresztül a vulkanológiai központba.

A vulkanológiai központ épületét egy 3D-ben tervezett és nyomtatott földrengésbiztos modell reprezentálja. A környezetében elhelyezett másik Microbit vezérlő feladata a mérőállomásról Bluetooth-on érkező vulkáni paraméterek fogadása, és a konzolon grafikon formájában történő megjelenítése, illetve kritikus érték esetén figyelmeztető jelzés küldése.

A Microbit vezérlők programozása a MakeCode editorban valósul meg blokkprogramozással, illetve MicroPython-ban MU editorral.

Nyilvánvaló, hogy vulkáni körülményeket nem tudunk produkálni, és mivel csak oktatási célú szenzorokkal rendelkezünk a jelenségek szimulációja során kénytelenek vagyunk igazodni szenzoraink képességeihez. A gázt propán-bután gázzal, a gőzt és a hőt vízforralással, a zajt vulkánkitörés hangjával, a rezgést a modell rezgésbe hozásával, a fényhatást nyilt lánggal szimuláljuk. Szimuláció közben figyeljük a vulkanológiai központban a grafikonok változásait.

Célok

- A természeti erőforrások és a környezet iránti felelősségtudat kialakítása.
- Az energiaátalakulásokkal kapcsolatos társadalmi, technikai problémákhoz való tudatos viszony kialakítása.
- Rendszerben és alternatívákban való gondolkodás fejlesztése
- Problémamegoldó képesség fejlesztése
- Kreatív technológia használat
- Algoritmizálási, adatelemzési készség fejlesztése

Módszerek

- Alkotópedagógia
- Modellezés
- Kísérletezés
- Projekt

Projekttevékenységek

Előkészítés, információgyűjtés

- Mit tudunk a vulkánokról?
- Mit gondoltak régen a vulkánokról? Mitológiai, tudománytörténeti vonatkozások (neptunista-vulkanista vita)
- Mi a geotermikus energia, hogyan hasznosítható?
- Hogyan lehetne hasznosítani a vulkáni energiákat?

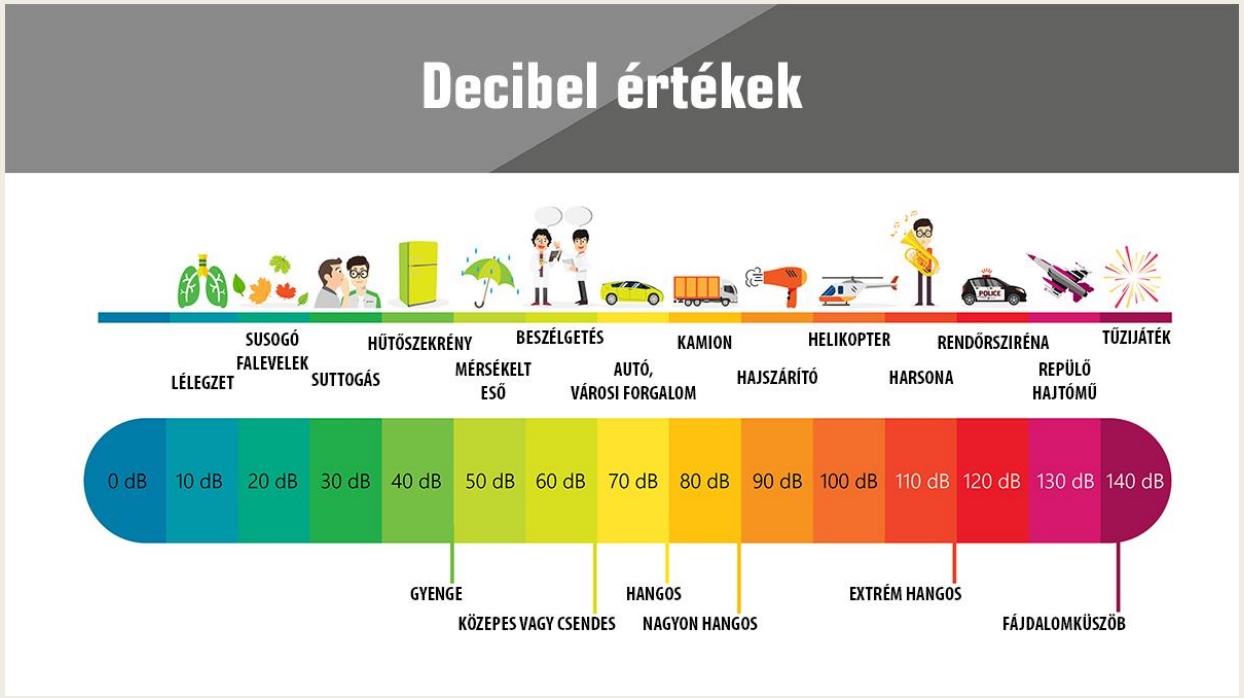
A leghangosabb hang, amit ember hallott

Krakatau 1883.

Hangerő elméleti határa 194 dB

160 km távolságban 180 dB

5000 km-re is hallható volt



Problémafelvetés

Vulkáni energiák hasznosítása során folyamatosan mérni szeretnénk a vulkán állapotát. Hogyan lehetne ezt megvalósítani?

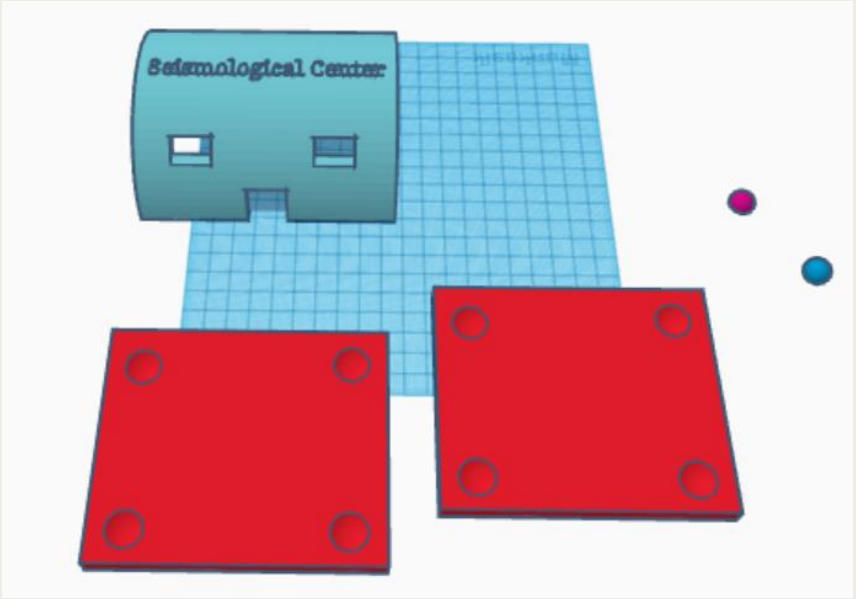
A rekordot jelentő, 4659 méteres mélységű 2017-es kutatófúrás eredményeképpen sikerült elérni a magmakamrát körülvevő kőzetet. Itt extrém nyomás alatt 427 °C-os vízhez fértek hozzá. Hőmérséklet szempontjából a csúcsot mégsem ez a fúrás jelenti. 2009-ben a program egyik első fúrása alkalmával véletlenül egyenesen a magmába ütköztek, majd stabilizálni is sikerült a geotermikus kitermelés során valaha előállított legforróbb, több mint 450 fokos gőzt termelő rendszert.

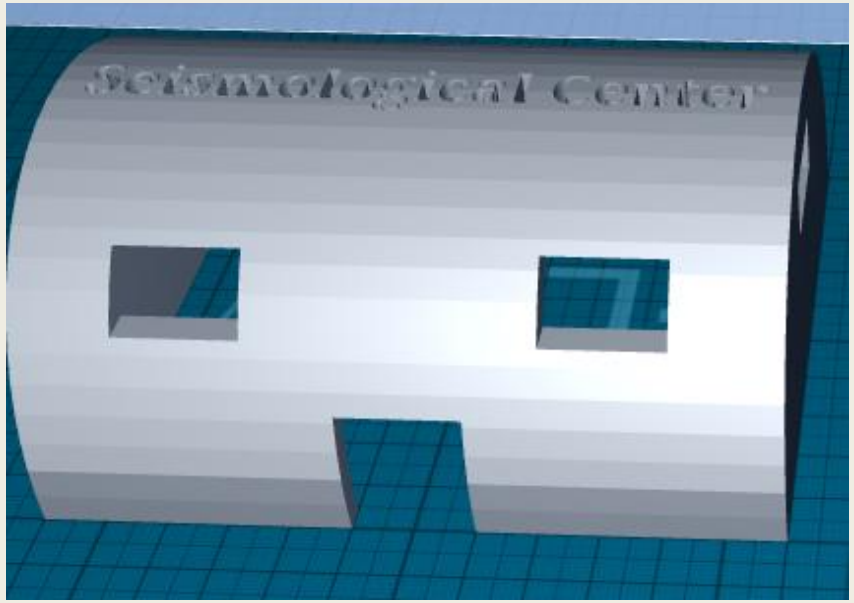
https://index.hu/techtud/2021/05/26/vulkan-kitores-hamu-klimavaltozas-ocean-izland/

Tevékenységek

Vulkán modell, mérőállomás elkészítése

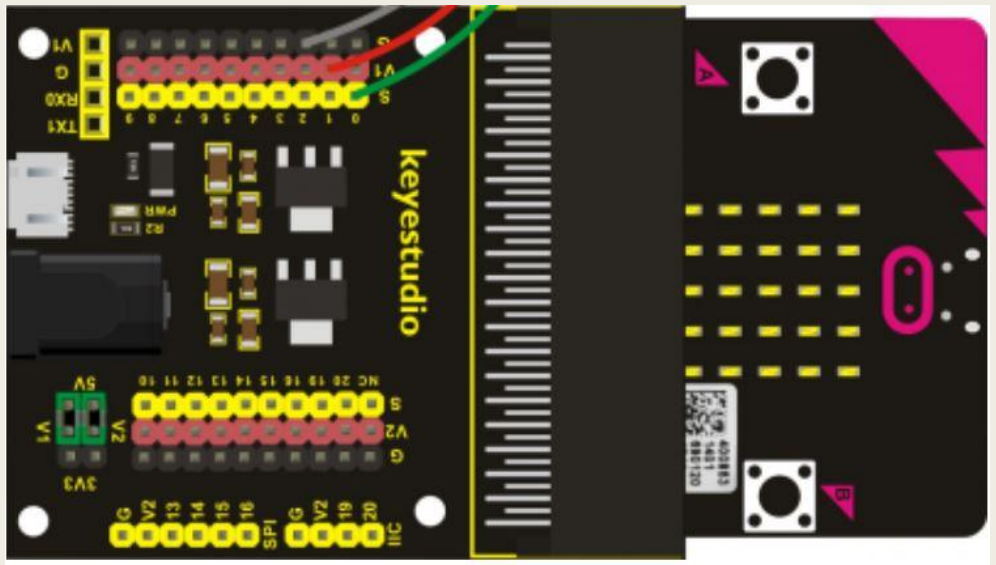
Földrengésbiztos vulkanológiai központ
3D tervezése, nyomtatása





Szenzorok kiválasztása, bekötése

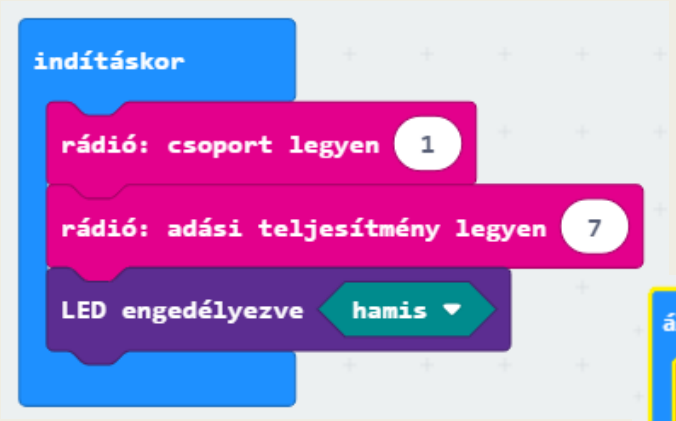
- P0 Gőzérzékelő
- P1 Hőérzékelő
- P2 Gázérzékelő
- P3 Rezgésérzékelő
- P4 Zajszint érzékelő
- P10 Tűzérzékelő



Szenzorok méréshatárai

paraméter hu	paraméter en	min.	max.	határérték
zaj	noise	10	400	100
rezgés	vibratio	2	1016	400
gáz	gas	40	800	500
gőz	vapor	70	680	400
tűz	fire	750	1000	950
hőmérséklet	temp	440	940	700

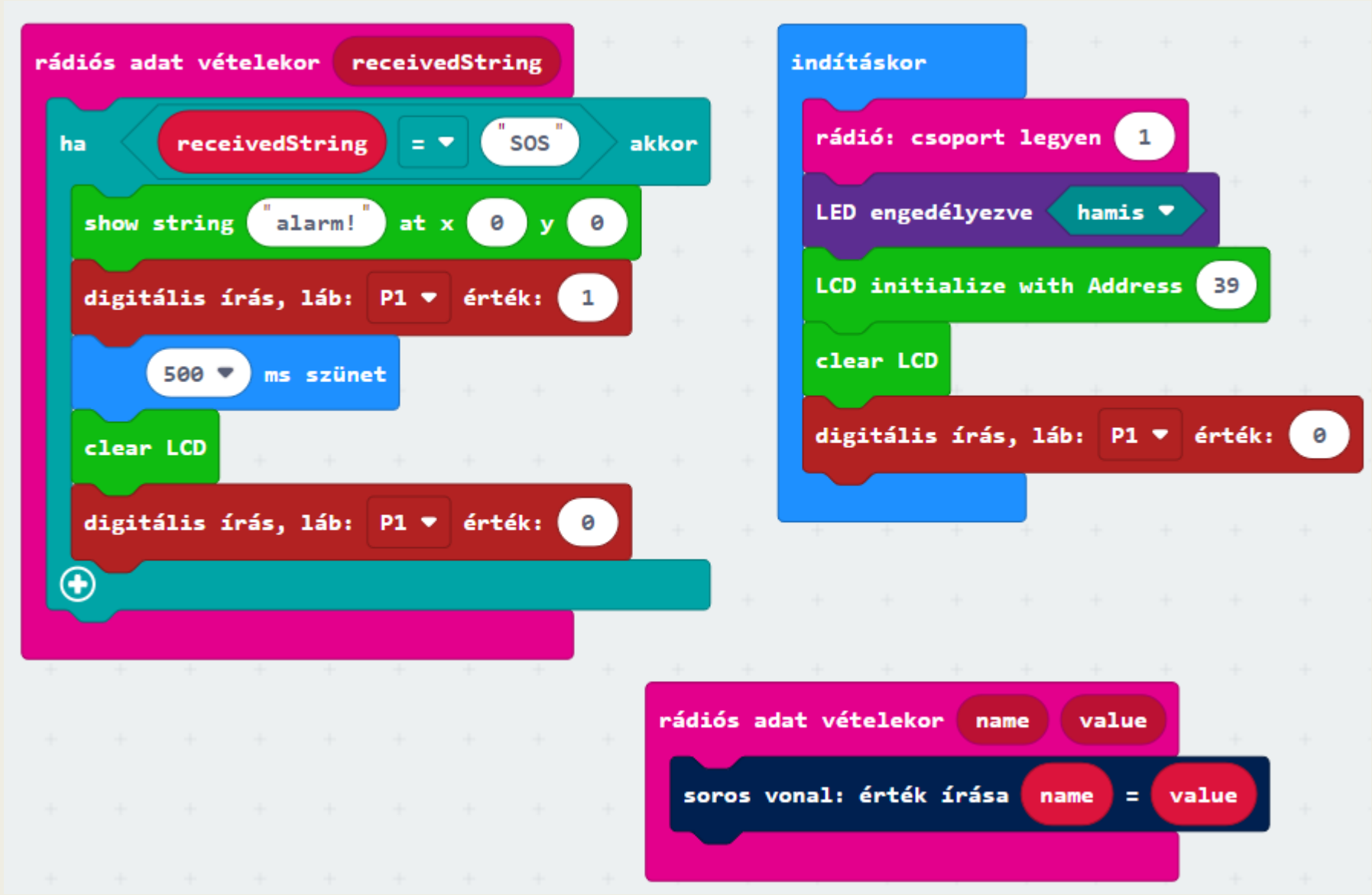
Mérőállomás programja



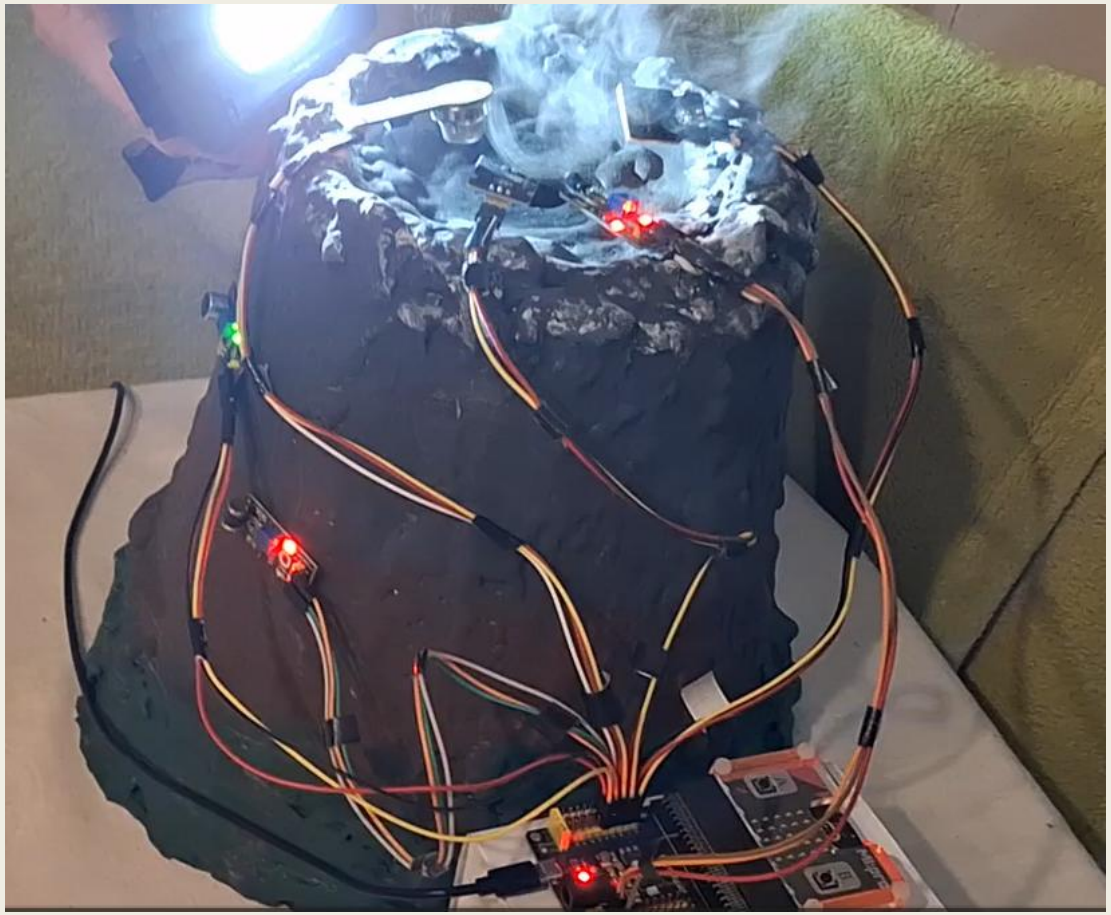
```python
# vulkáni paraméterek küldése Microbit A

from microbit import *
import radio

radio.config(group = 1, power = 7)
radio.on()

while True:
    parameter_nev = 'goz'
    parameter_ertek = pin0.read_analog()
    uzenet = parameter_nev + str(parameter_ertek)
    radio.send(uzenet)
    sleep(100)

    parameter_nev = 'hom'
    parameter_ertek = pin1.read_analog()
    uzenet = parameter_nev + str(parameter_ertek)
    radio.send(uzenet)
    sleep(100)

    parameter_nev = 'gaz'
    parameter_ertek = pin2.read_analog()
    uzenet = parameter_nev + str(parameter_ertek)
    radio.send(uzenet)
    sleep(100)
```

```python
while True:
    if button_a.was_pressed():
        radio.config(group = 1)

    if button_b.was_pressed():
        radio.config(group = 2)

    uzenet = radio.receive()

    if uzenet:

        parameter_nev = uzenet[0:3]
        parameter_ertek = int(uzenet[3: ])

        if parameter_nev == 'goz':
            goz_ertek = parameter_ertek

        if parameter_nev == 'hom':
            hom_ertek = parameter_ertek

        if parameter_nev == 'gaz':
            gaz_ertek = parameter_ertek

        if parameter_nev == 'rez':
            rez_ertek = parameter_ertek

        if parameter_nev == 'zaj':
            zaj_ertek = parameter_ertek
```

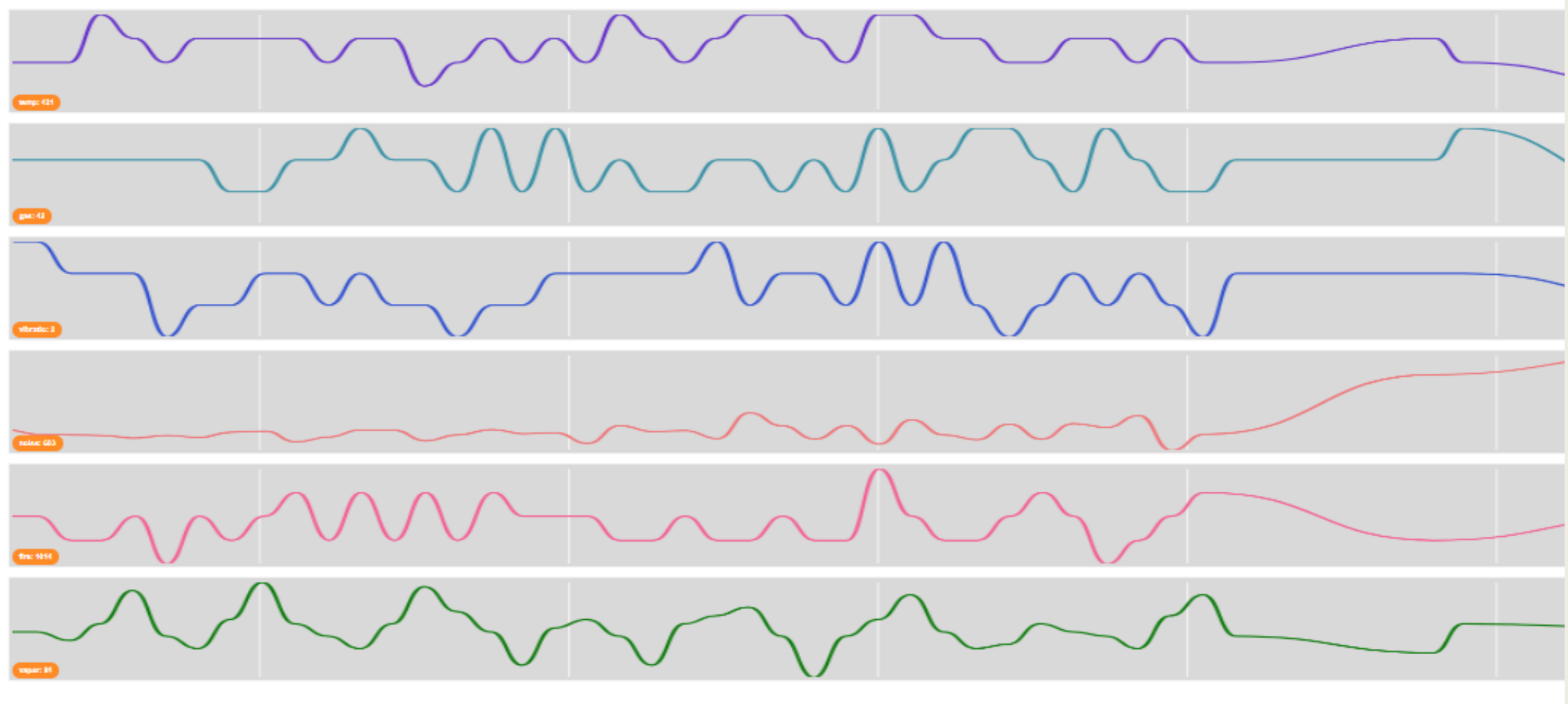
Vulkanológiai központ programja



Mérési folyamat modellezése, tesztelés



A mért adatok grafikonon



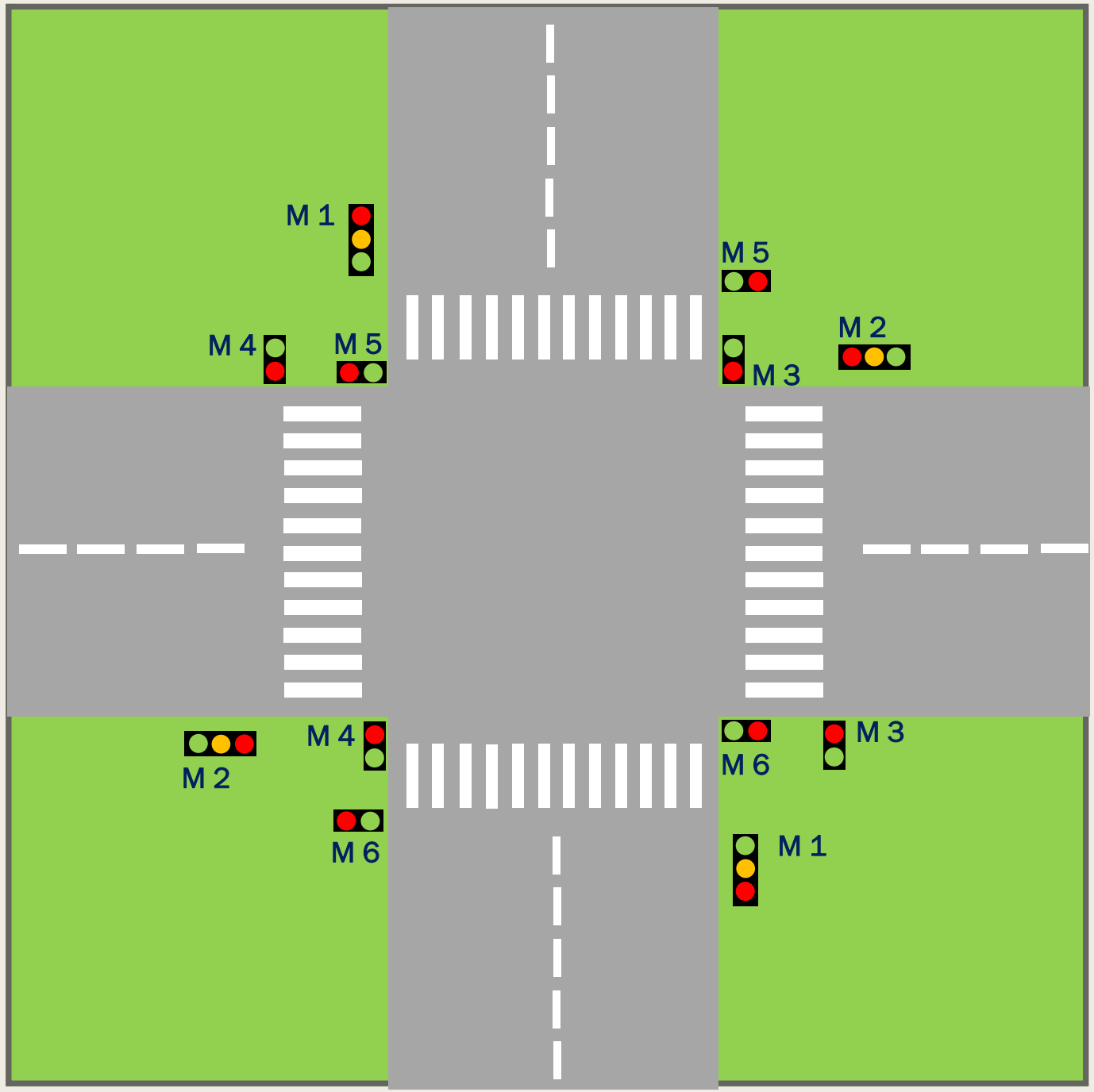
A mért adatok táblázatban

time (forrÃ˘s1)	**temp**	time (forrÃ˘s1)	**gas**	time (forrÃ˘s1)	**vibratio**	time (forrÃ˘s1)	**noise**	time (forrÃ˘s1)	**fire**	time (forrÃ˘s1)	**vapor**
	0		0		0		0		0		0
	424		41		2		619		1012		97
3.104	423	3.099	40	3.576	2	3.698	645	3.699	1013	0.48	100
3.586	425	3.583	41	7.269	2	7.385	629	7.385	1014	3.699	101
7.286	424	7.282	41	11.488	3	8.414	600	8.416	1014	7.387	108
10.492	424	10.492	41	12.014	3	12.137	633	12.137	1013	7.878	91
11.505	424	11.502	40	19.404	2	19.531	634	19.531	1014	8.923	101
15.24	424	15.236	41	26.796	3	26.911	628	26.913	1013	12.621	95
15.727	424	15.728	41	30.494	0	30.611	624	30.612	1014	15.835	105
18.939	425	18.934	41	34.191	2	34.313	624	34.314	1013	16.315	86
22.635	424	22.631	41	41.58	1	41.703	651	41.703	1012	20.011	98
23.118	424	23.114	42	45.278	3	45.403	662	45.406	1014	23.228	94
26.332	423	26.327	42	48.962	1	45.885	509	45.886	1013	23.707	104
26.814	423	26.809	41	49.483	2	46.411	497	46.411	1013	26.914	87

Közlekedési projekt

Közlekedési lámpák működésének modellezése útkereszteződésben

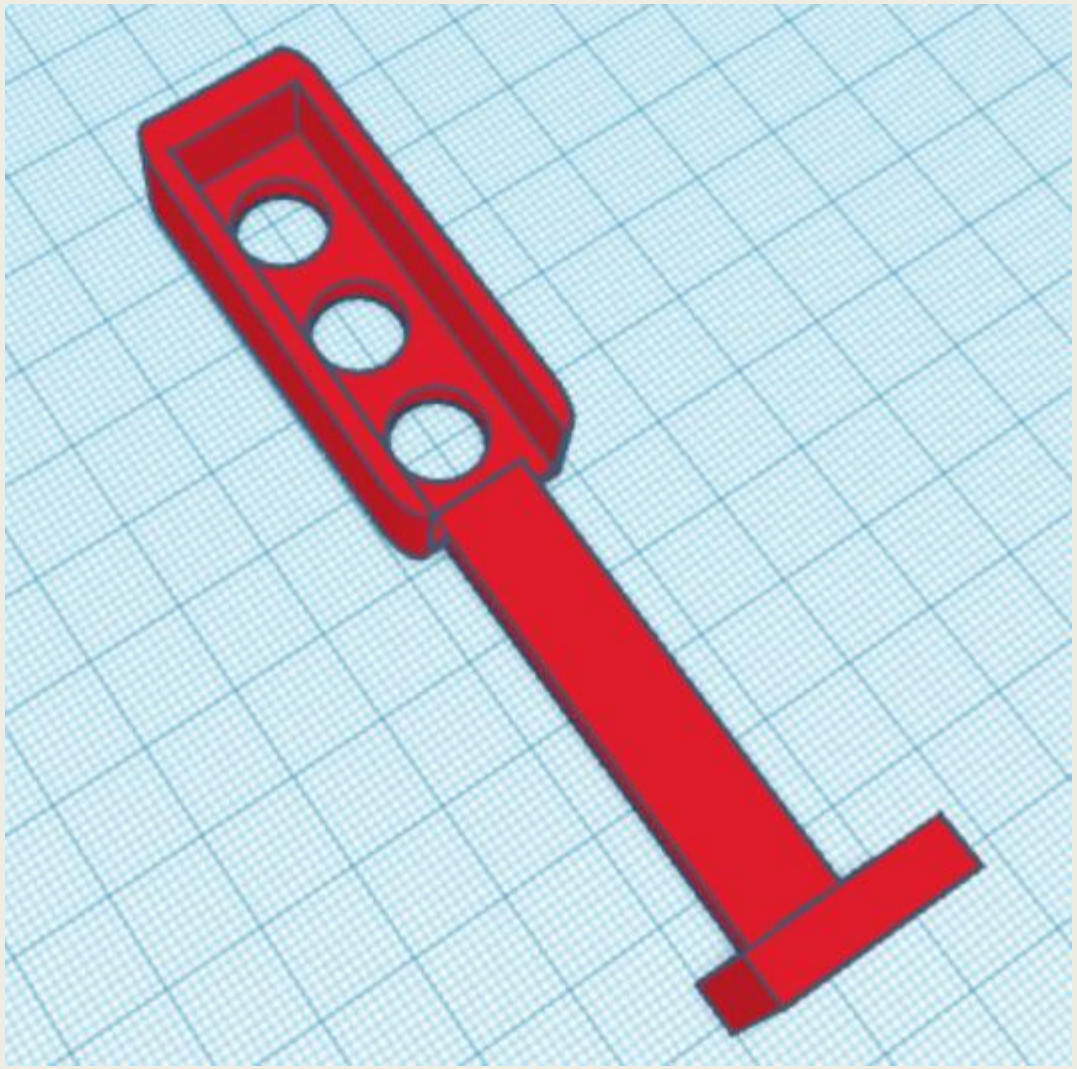
A modell egy útkereszteződés lámpáinak összehangolt működését valósítja meg 12 közlekedési lámpával, 6 programozott Microbit vezérlővel, Bluetooth kommunikációval. A modell fizikai alapja egy 60 cm*60 cm*3 cm-es lépésálló hungarocell tábla. Erre kerül felfestésre az útkereszteződés, a 3D-ben tervezett, nyomtatott közlekedési lámpák, elektronika, programozott Microbit vezérlők.

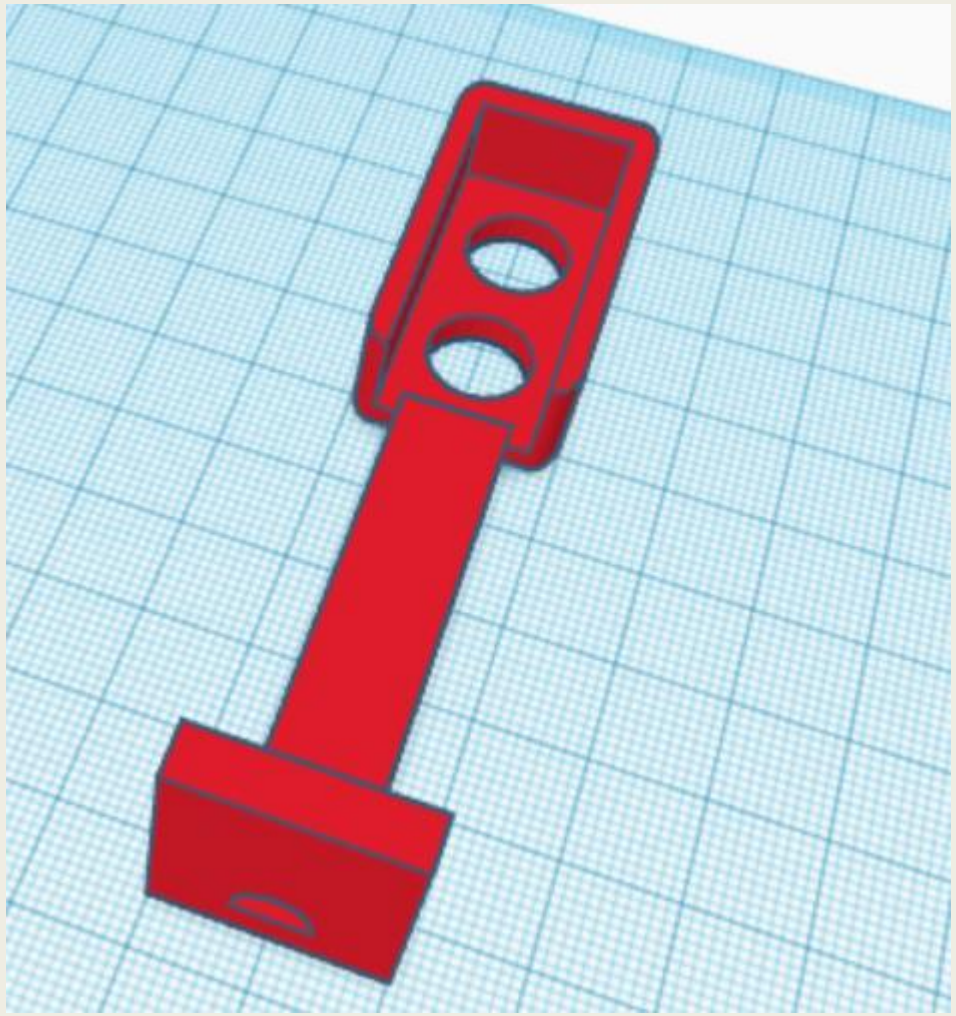


M 1, M 2 Microbit vezérli a szemközti forgalmi irányú két-két jármű közlekedési lámpát.

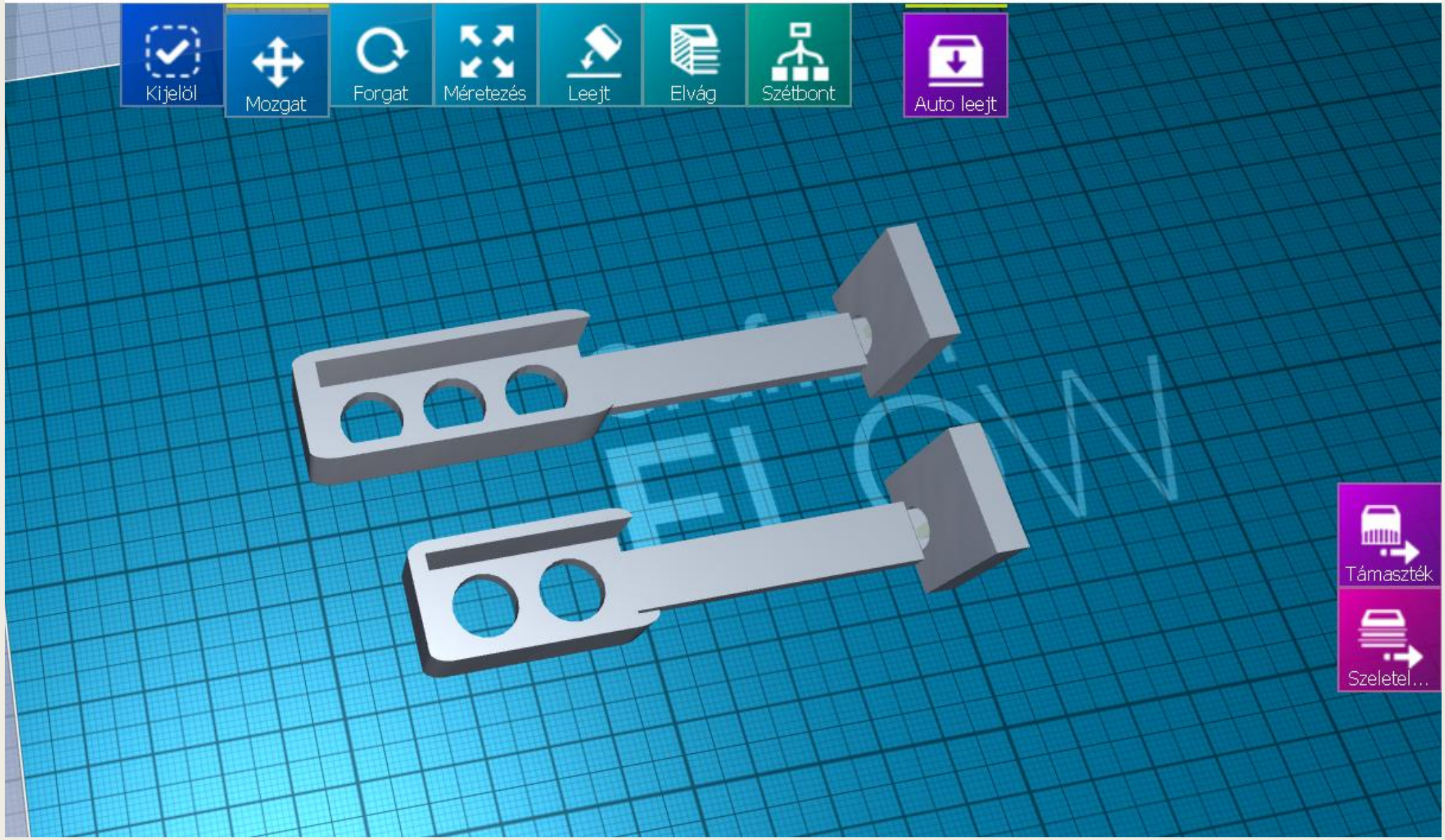
M 3, M 4, M 5, M 6 Microbit vezérli a szemközti forgalmi irányú gyalogos közlekedési lámpákat.

Közlekedési lámpák 3D terve



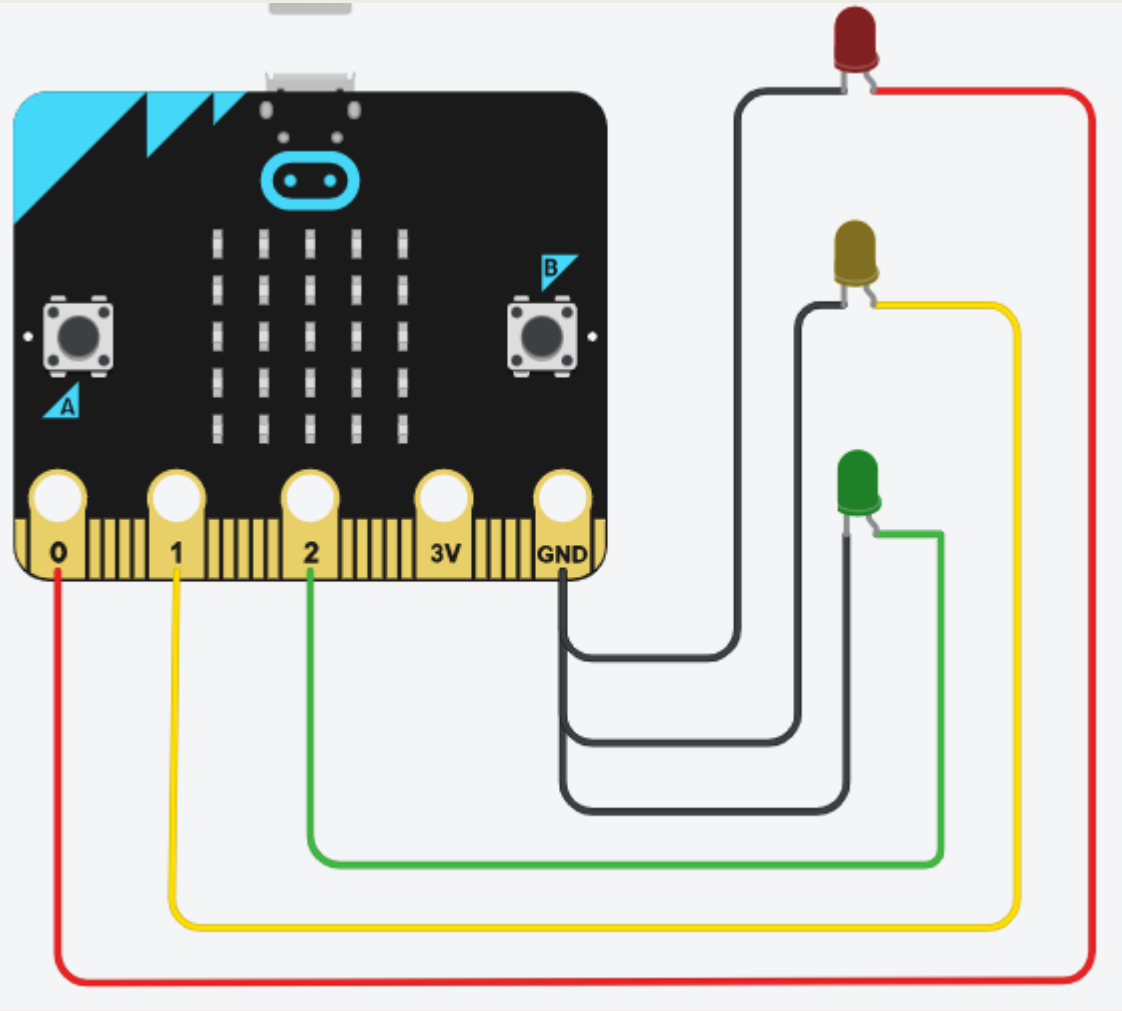


3D nyomtatás előkészítése

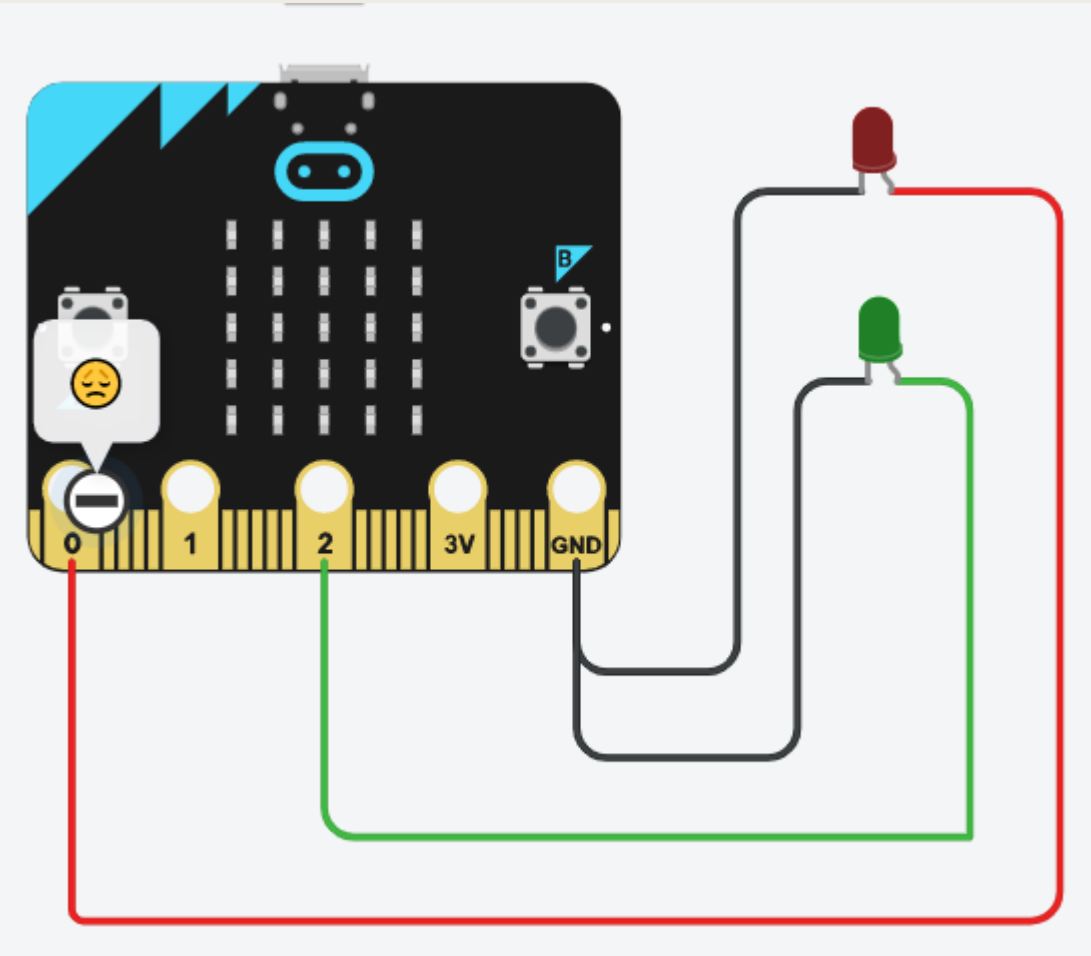


Közlekedési lámpák kapcsolási rajza

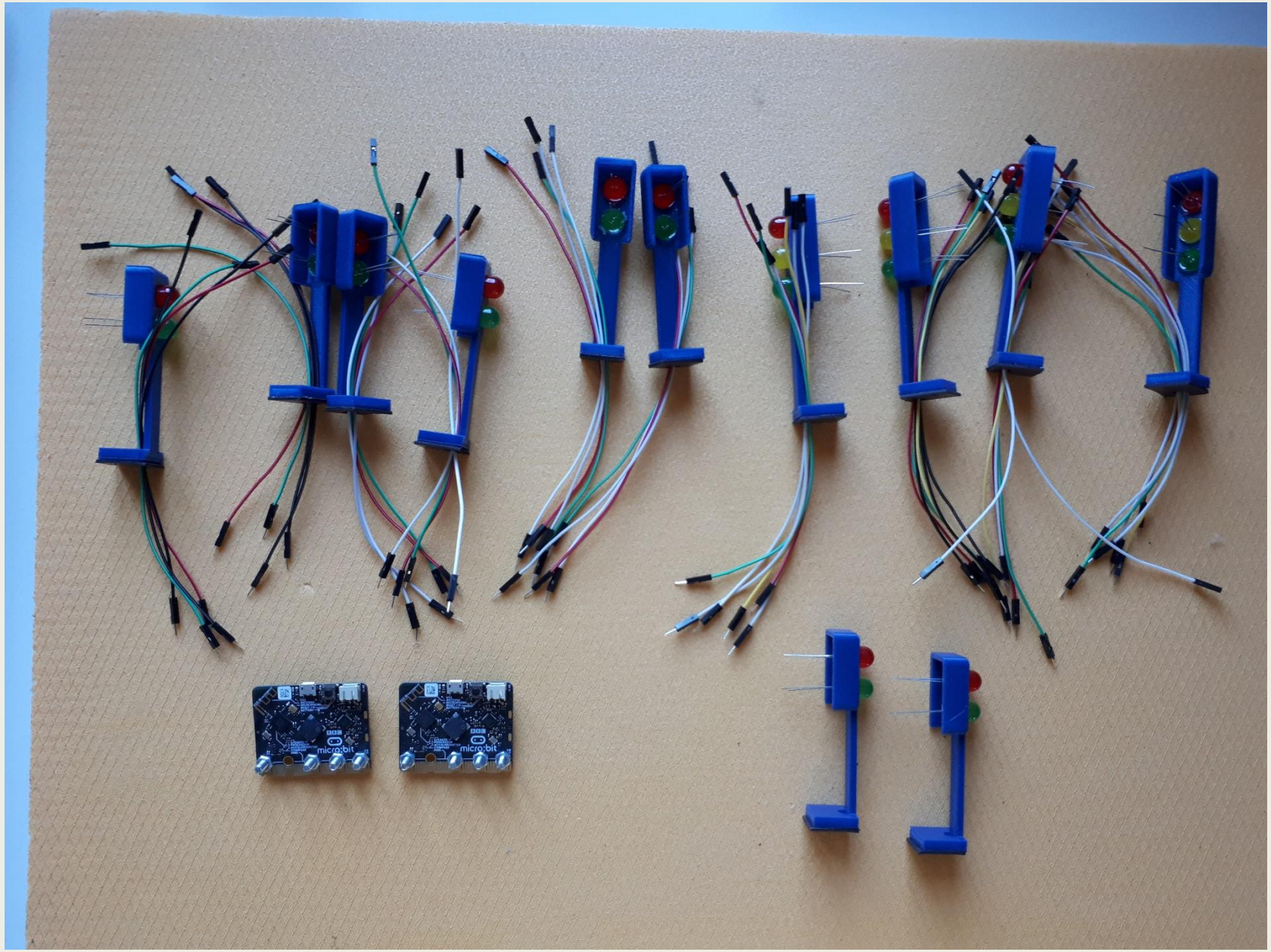
jármű



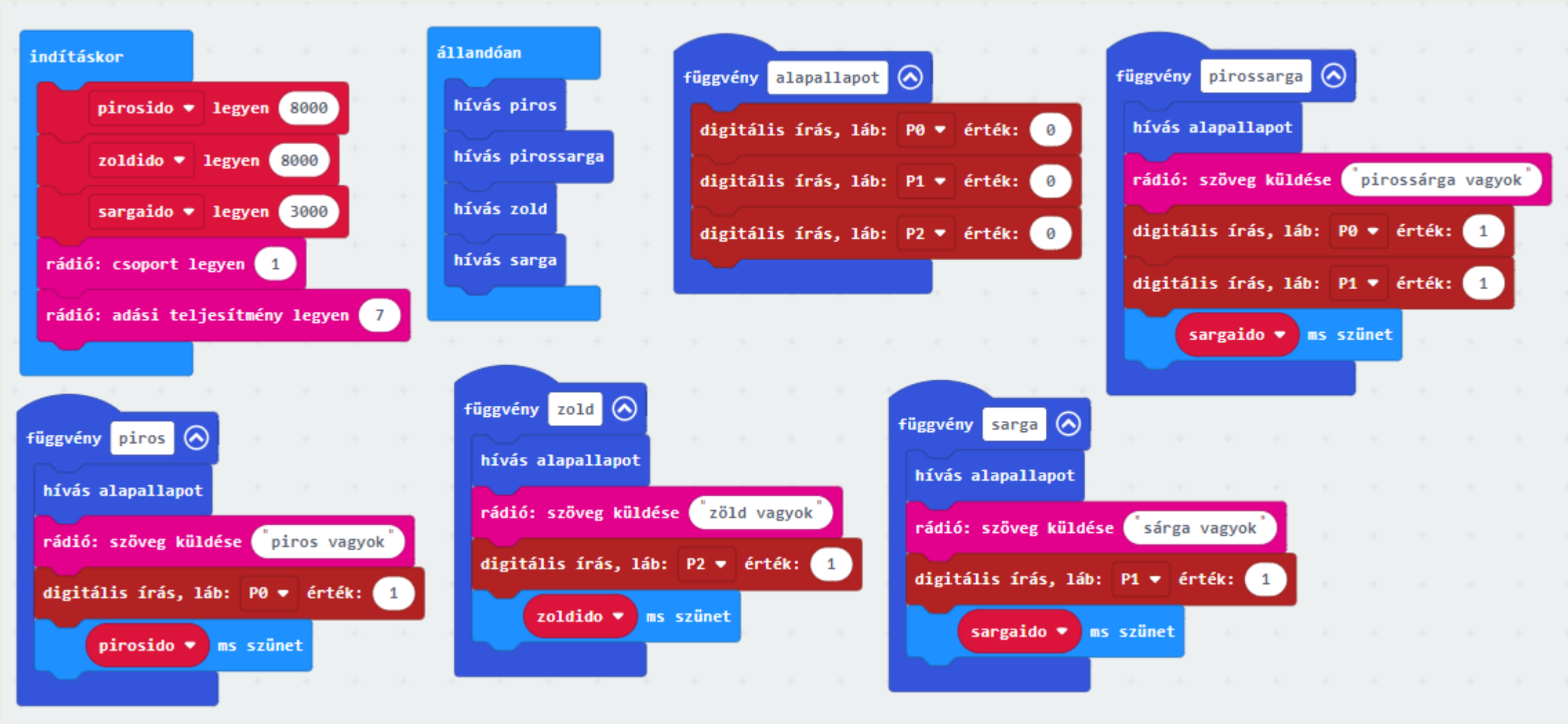
gyalogos



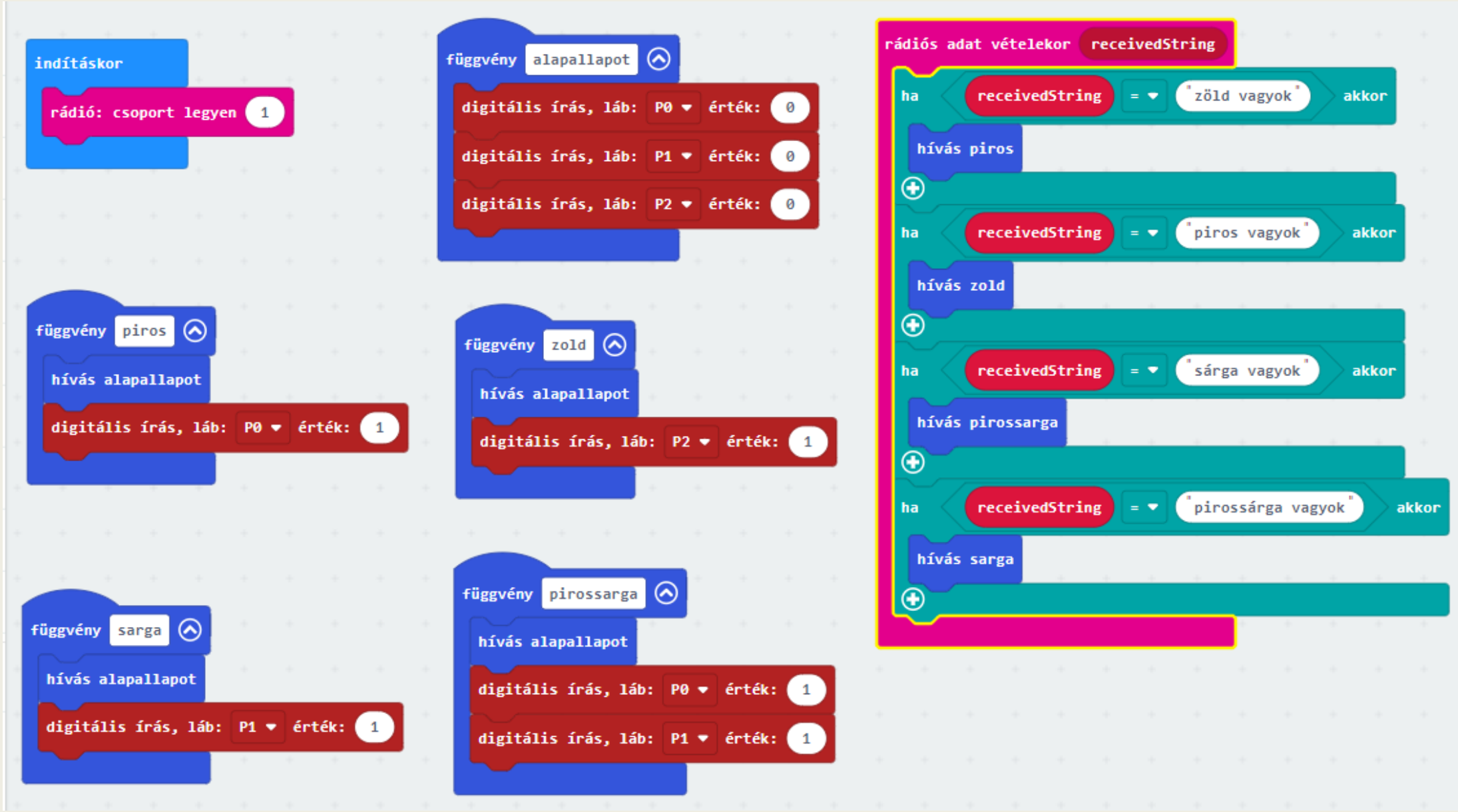
Ledek, vezetékek bekötése a lámpatestbe, Microbitre



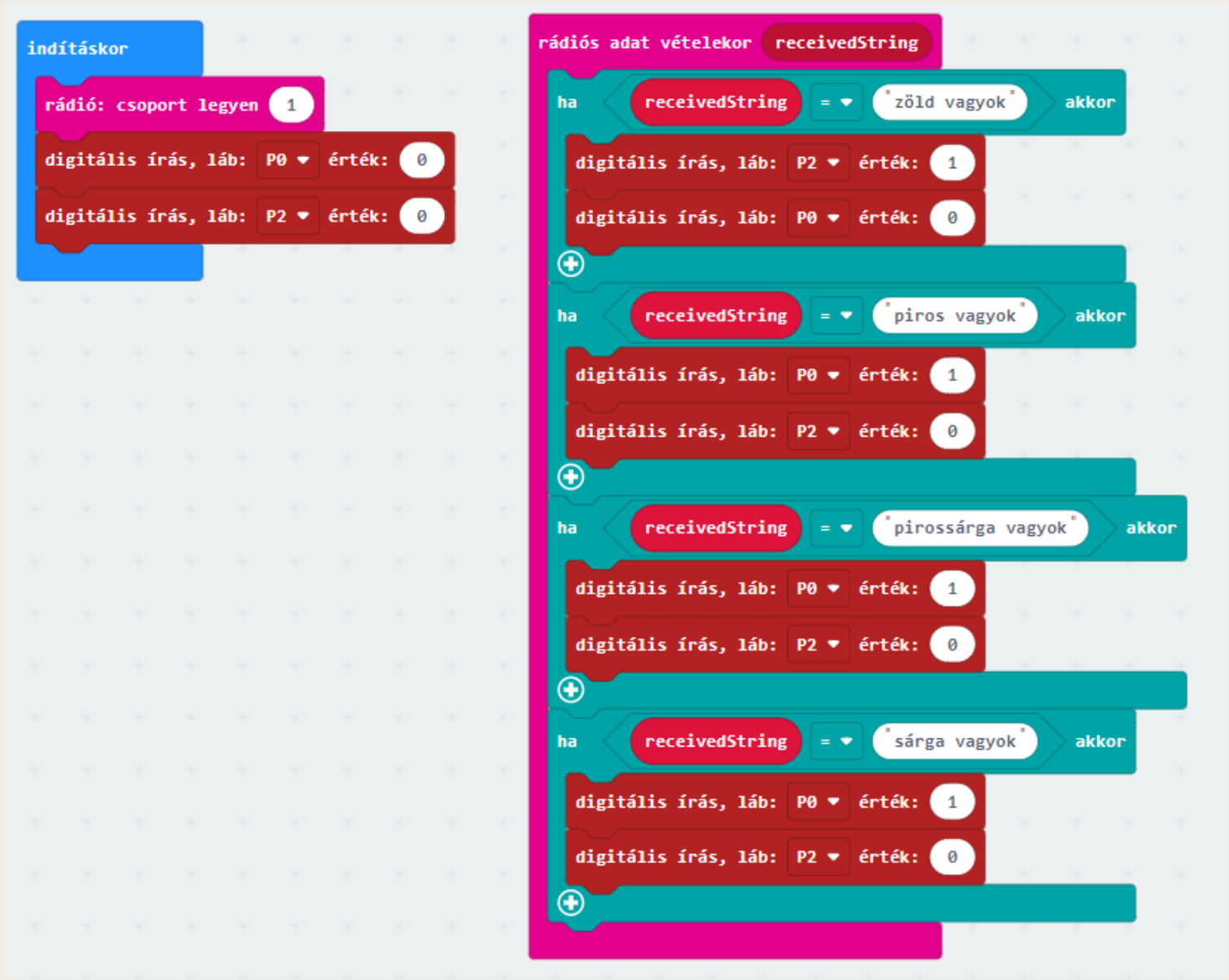
M1 Microbit programkódja (két szemben lévő jármű lámpát irányítja, állapotait Bluetooth-on folyamatosan küldi a többi lámpának)



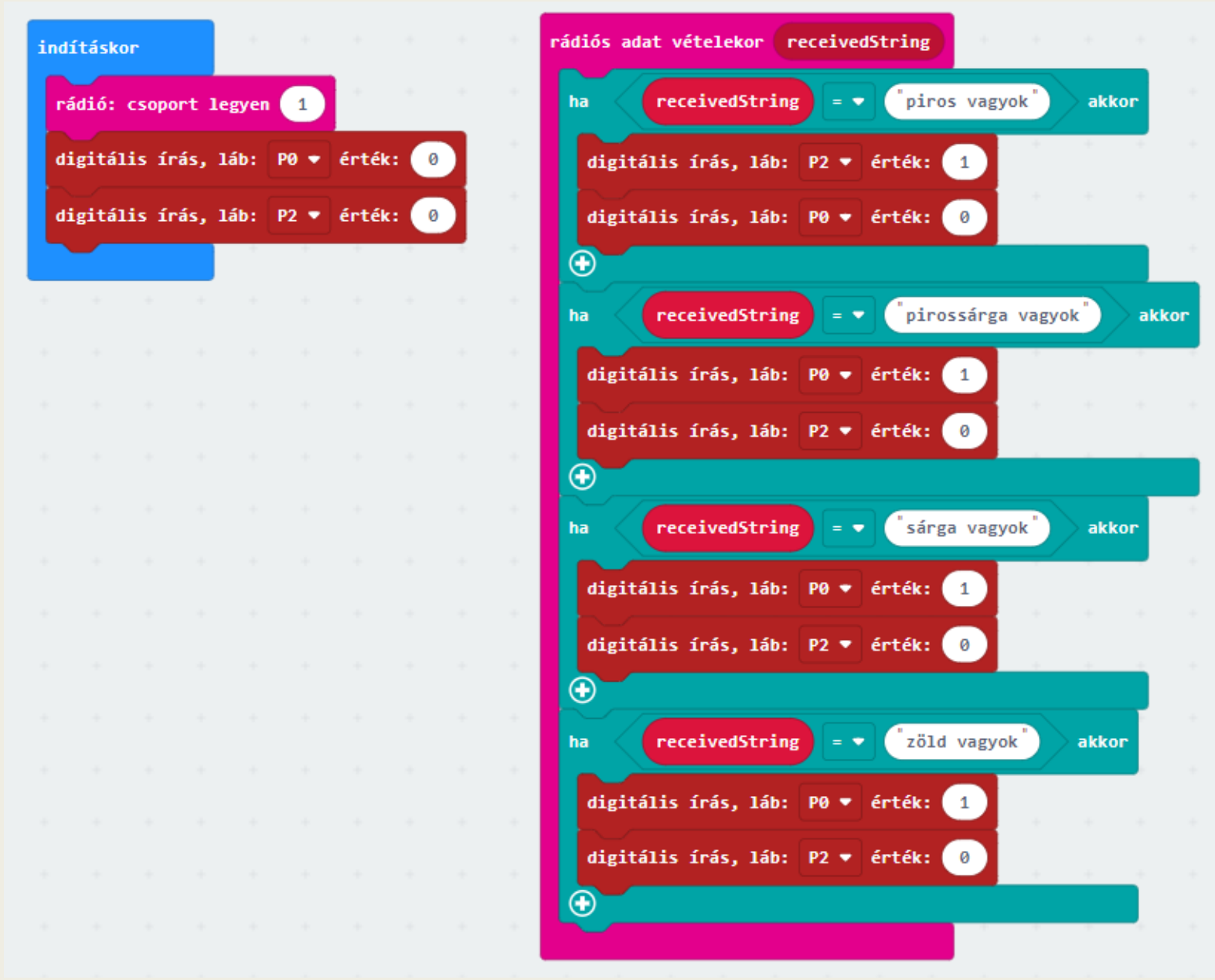
M2 Microbit programkódja (másik két szemben lévő jármű lámpát irányítja M1-től kapott állapotok alapján)



M3 M4 Microbit programkódja (2-2 egyformán működő gyalogos lámpát irányít az M1-től kapott állapotok alapján.



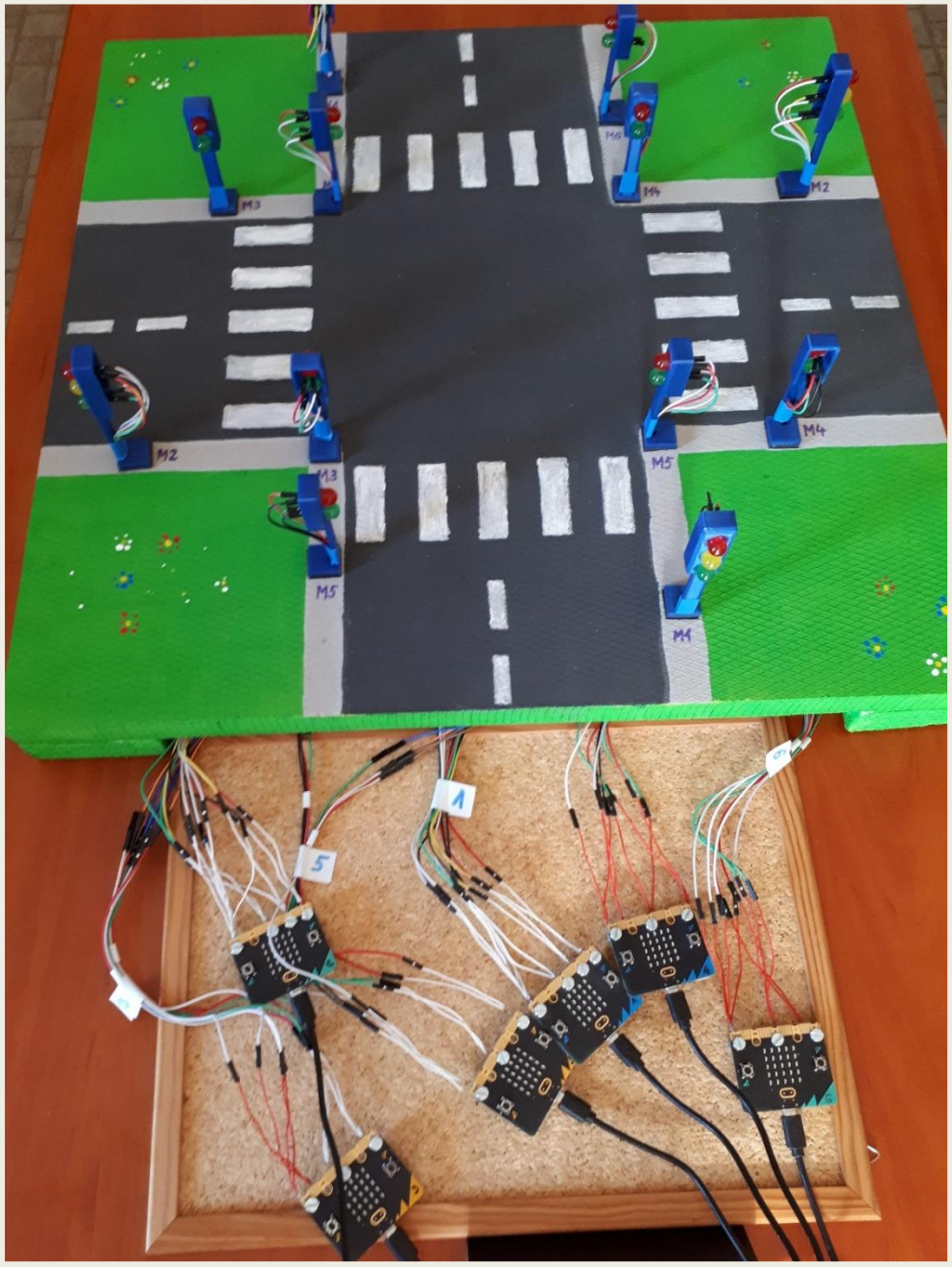
M5 M6 Microbit programkódja (2-2 egyformán működő gyalogos lámpát irányít az M1-től kapott állapotok alapján.



```
# microbit-útkereszteződés-jármű-M2

from microbit import *
import radio

radio.config(group = 1)
radio.on()

def alapallapot():
    pin0.write_digital(0)
    pin1.write_digital(0)
    pin2.write_digital(0)

def piros():
    alapallapot()
    pin0.write_digital(1)

def zold():
    alapallapot()
    pin2.write_digital(1)

def sarga():
    alapallapot()
    pin1.write_digital(1)

def pirossarga():
    alapallapot()
    pin0.write_digital(1)
    pin1.write_digital(1)
```

```
# microbit-útkereszteződés-gyalogos-M5-M6_v1

from microbit import *
import radio

radio.config(group = 1)
radio.on()

while True:
    uzenet = radio.receive()

    if uzenet:

        if uzenet = ('piros vagyok'):
            pin2.write_digital(1)
            pin0.write_digital(0)

        if uzenet = ('pirossarga vagyok'):
            pin0.write_digital(1)
            pin2.write_digital(0)

        if uzenet = ('sarga vagyok'):
            pin0.write_digital(1)
            pin2.write_digital(0)

        if uzenet = ('zold vagyok'):
            pin0.write_digital(1)
            pin2.write_digital(0)
```

Tesztelés



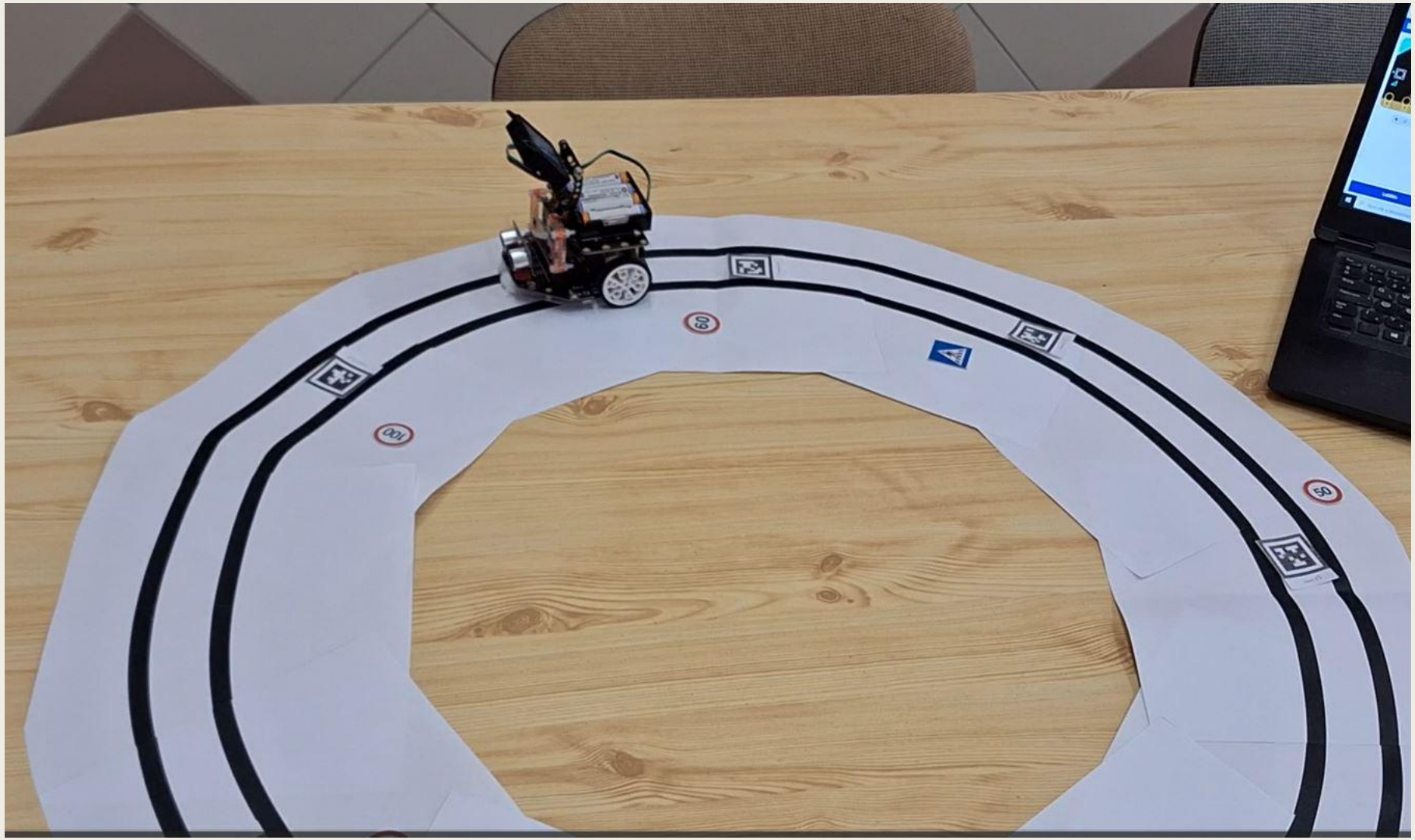
Önvezető jármű modellezése mesterséges intelligencia kamerával

Modell bemutatása

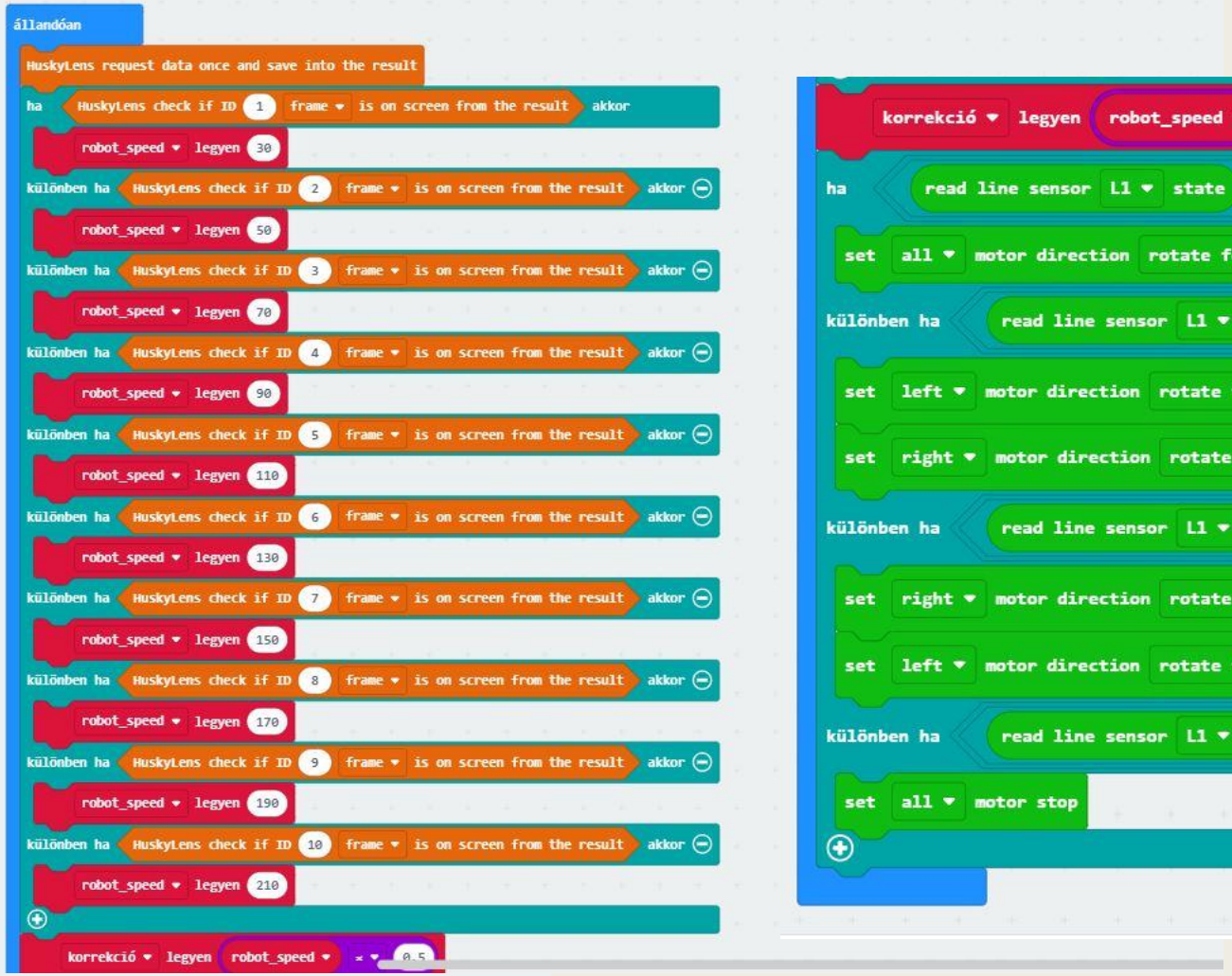
A Huskylens gépi látás érzékelőt betanítottuk cimkék felismerésére. Ezek a cimkék mindegyike egy-egy sebességkorlátozó táblát reprezentál. A cimkéket biztosabban ismeri fel a mesterséges intelligencia algoritmus, mint a közlekedési táblákat. Az az elképzelésünk, hogy az útburkolatba be lehetne építeni led mátrixokat, amelyek távolról programozhatók. Ezek reprezentálnák a cimkéket. Így sebességkorlátozás változása esetén nem kellene táblákat cserélni. Az önvezető autók egyik kamerája ezeket a cimkéket olvasná, és az elektronika leszabályozná az autó sebességét. Így elkerülhető lenne a gyorshajtás, amely a közlekedési balesetek nagyrészének az oka.

A modell egy Microbit által vezérelt Maqueen V2 Plus robot és a Huskylens gépi látásérzékelő együttműködésére épül. A robotot úgy programozzuk, hogy a különböző cimkék felismerésekor változtassa a sebességét a cimkének megfeleltetett sebességre.

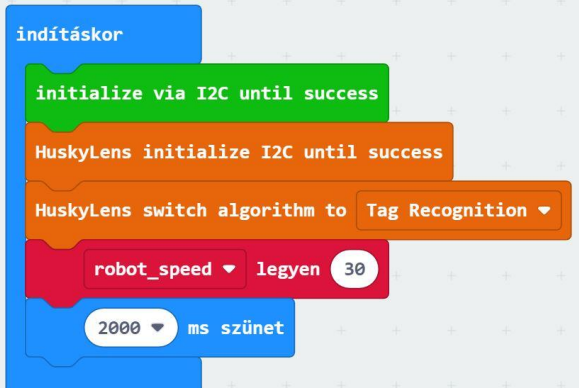
Programozásra a microbit.org online fejlesztői környezetet használjuk a Maqueen robot és a Huskylens gépi látásérzékelő bővítményeinek betöltésével.



Magyar nyelvű videó:
https://youtu.be/w2DfyCy37b8


állandóan
HuskyLens request data once and save into the result
ha HuskyLens check if ID 1 frame is on screen from the result akkor
robot_speed legyen 30
különben ha HuskyLens check if ID 2 frame is on screen from the result akkor
robot_speed legyen 50
különben ha HuskyLens check if ID 3 frame is on screen from the result akkor
robot_speed legyen 70
különben ha HuskyLens check if ID 4 frame is on screen from the result akkor
robot_speed legyen 90
különben ha HuskyLens check if ID 5 frame is on screen from the result akkor
robot_speed legyen 110
különben ha HuskyLens check if ID 6 frame is on screen from the result akkor
robot_speed legyen 130
különben ha HuskyLens check if ID 7 frame is on screen from the result akkor
robot_speed legyen 150
különben ha HuskyLens check if ID 8 frame is on screen from the result akkor
robot_speed legyen 170
különben ha HuskyLens check if ID 9 frame is on screen from the result akkor
robot_speed legyen 190
különben ha HuskyLens check if ID 10 frame is on screen from the result akkor
robot_speed legyen 210
korrekció legyen robot_speed × 0.5


korrekció legyen robot_speed × 0.5
ha read line sensor L1 state = 0 és read line sensor R1 state = 0 akkor
set all motor direction rotate forward speed robot_speed
különben ha read line sensor L1 state = 1 és read line sensor R1 state = 0 akkor
set left motor direction rotate forward speed robot_speed + korrekció
set right motor direction rotate forward speed robot_speed - korrekció
különben ha read line sensor L1 state = 0 és read line sensor R1 state = 1 akkor
set right motor direction rotate forward speed robot_speed + korrekció
set left motor direction rotate forward speed robot_speed - korrekció
különben ha read line sensor L1 state = 1 és read line sensor R1 state = 1 akkor
set all motor stop


indításkor
initialize via I2C until success
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Tag Recognition
robot_speed legyen 30
2000 ms szünet

Arcfelismerésen alapuló beléptetőrendszer modellezése

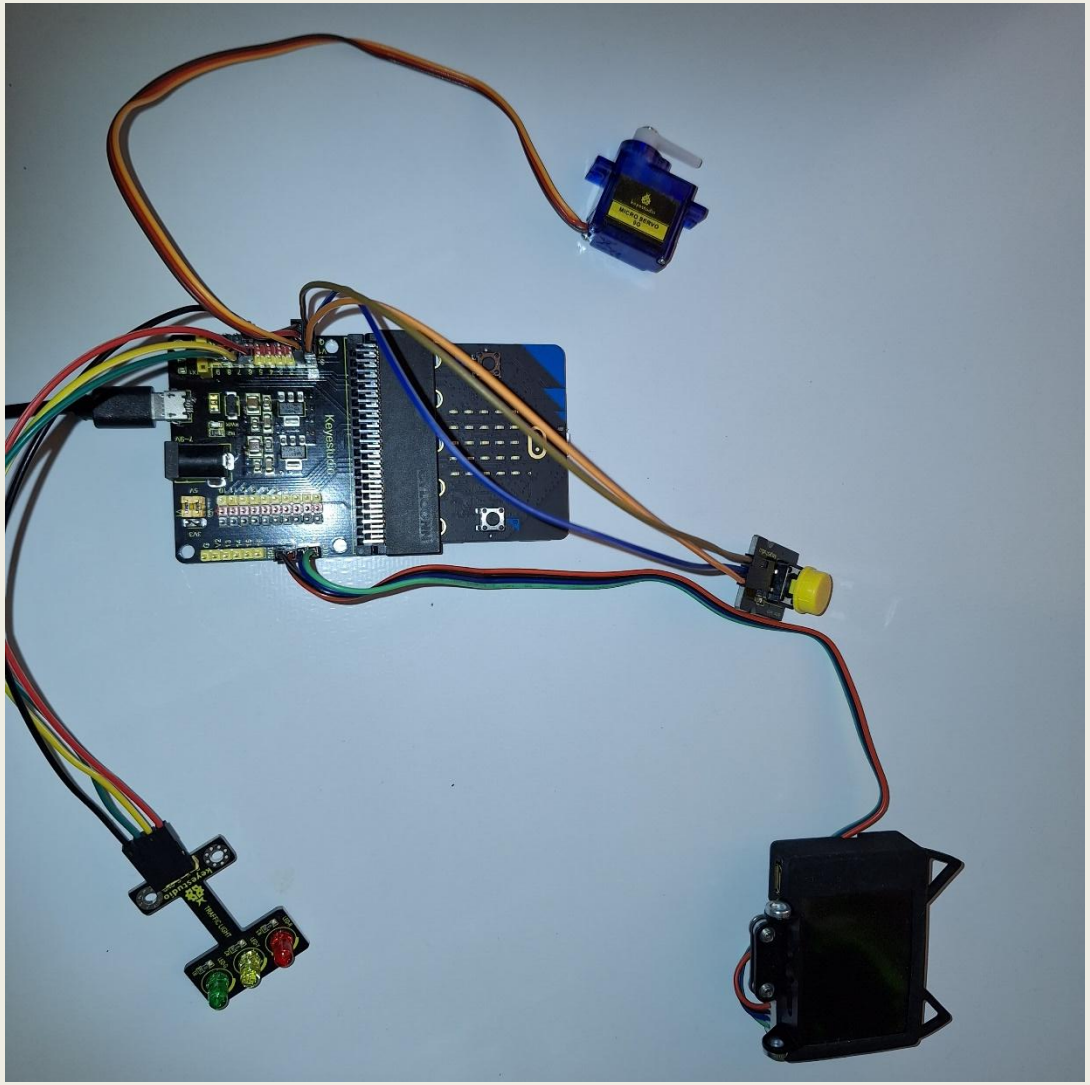
Modell bemutatása

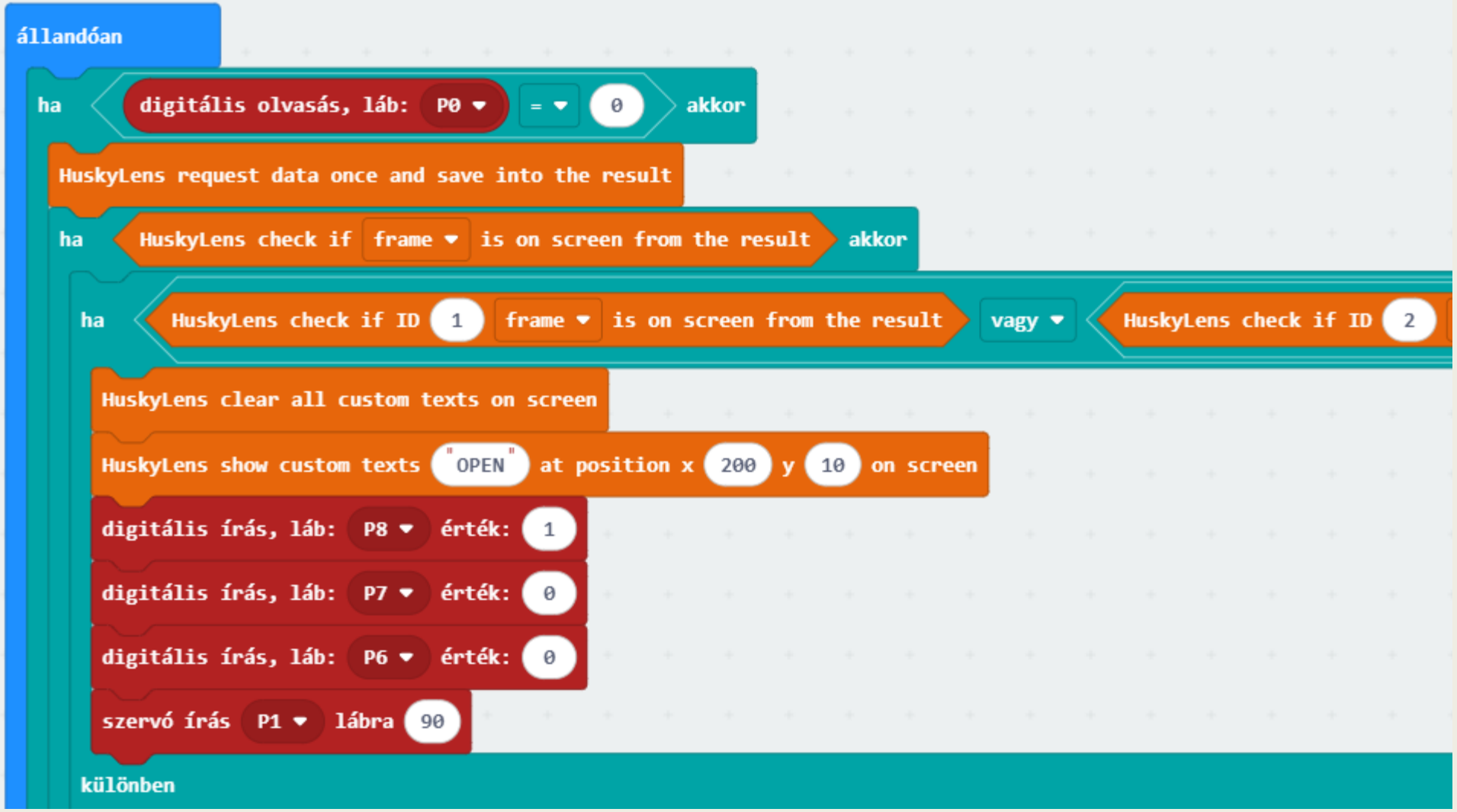
Családi ház arcfelismerésen alapuló beléptetőrendszerét modellezzük Microbit vezérlővel és Huskylens gépi látásérzékelő kamerával. A látásérzékelő beépített arcfelismerő gépi tanulás algoritmusát alkalmazva betanítjuk családtagok arcának felismerésére. Ezután az eszközt a Microbit vezérlőn keresztül úgy programozzuk, hogy a tanult modellt használja azonosításra, amikor valaki belenéz a kamerába. Csak a családtagok számára nyitja ki a 3D tervezett ház ajtaját egy szervo motor.

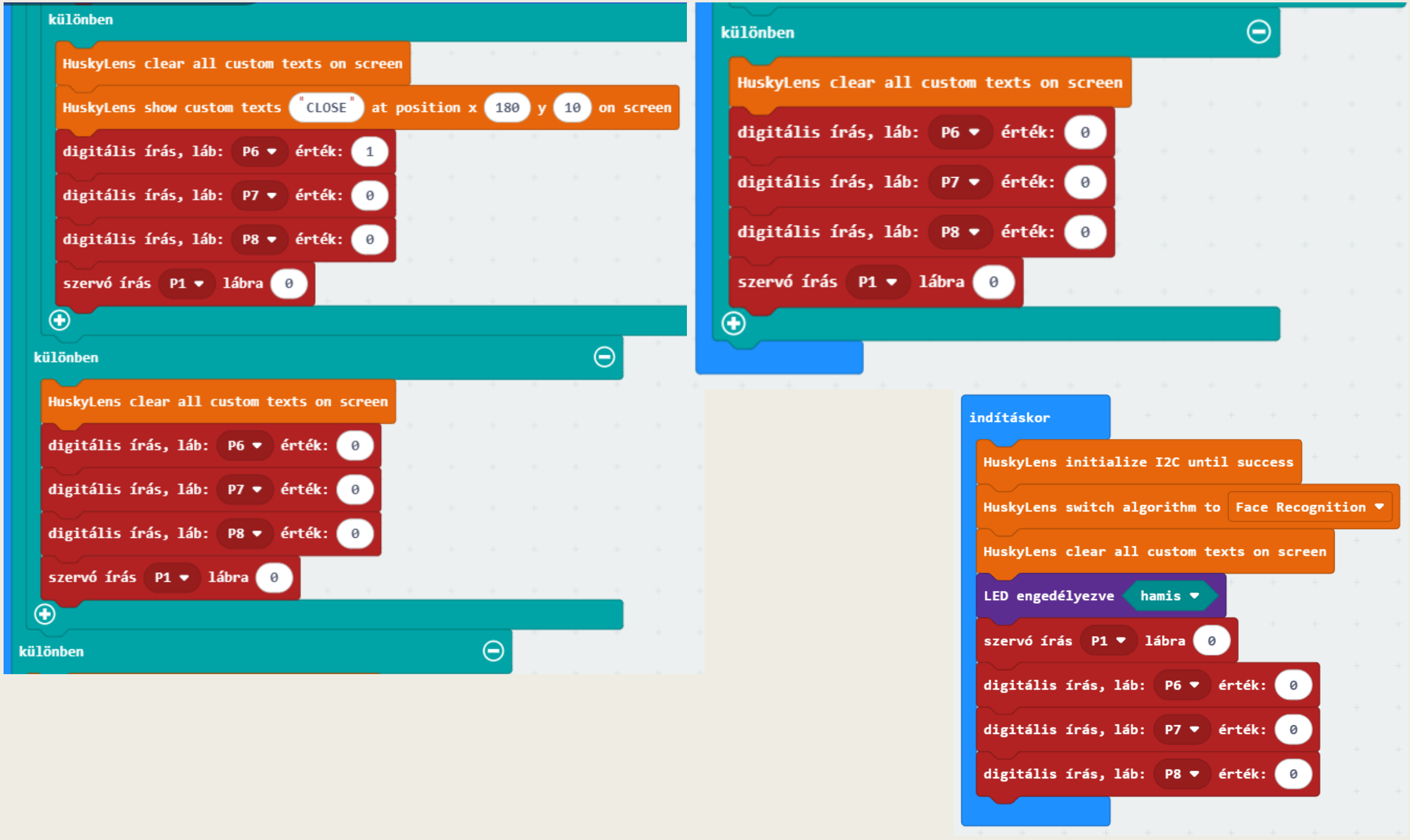
PIN kiosztás a Microbit vezérlő bővítőkártyáján

bemenet-kimenet	PIN
Huskylens	IIC
nyomógomb	P0
servo	P1
LED	R_P6
LED	Y_P7
LED	G_P8

Elektronikai elemek installációja




állandóan
ha
digitális olvasás, láb: P0
=
0
akkor
HuskyLens request data once and save into the result
ha
HuskyLens check if frame is on screen from the result
akkor
ha
HuskyLens check if ID 1 frame is on screen from the result
vagy
HuskyLens check if ID 2
HuskyLens clear all custom texts on screen
HuskyLens show custom texts "OPEN" at position x 200 y 10 on screen
digitális írás, láb: P8 érték: 1
digitális írás, láb: P7 érték: 0
digitális írás, láb: P6 érték: 0
szervó írás P1 lábra 90
különben


különben
HuskyLens clear all custom texts on screen
HuskyLens show custom texts "CLOSE" at position x 180 y 10 on screen
digitális írás, láb: P6 érték: 1
digitális írás, láb: P7 érték: 0
digitális írás, láb: P8 érték: 0
szervó írás P1 lábra 0
különben
HuskyLens clear all custom texts on screen
digitális írás, láb: P6 érték: 0
digitális írás, láb: P7 érték: 0
digitális írás, láb: P8 érték: 0
szervó írás P1 lábra 0
különben
különben
HuskyLens clear all custom texts on screen
digitális írás, láb: P6 érték: 0
digitális írás, láb: P7 érték: 0
digitális írás, láb: P8 érték: 0
szervó írás P1 lábra 0
indításkor
HuskyLens initialize I2C until success
HuskyLens switch algorithm to Face Recognition
HuskyLens clear all custom texts on screen
LED engedélyezve hamis
szervó írás P1 lábra 0
digitális írás, láb: P6 érték: 0
digitális írás, láb: P7 érték: 0
digitális írás, láb: P8 érték: 0