37

RANGER POTENTIOMETER ARDUINO

STEMJAM Teaching Guide

Developing make spaces to promote creativity around STEM in schools

Acronym: STEMJAM

Project no. 2016-1-ES01-KA201-025470

www.stemjam.eu

Co-funded by the Erasmus+ Programme of the European Union

RANGER POTENTIOMETER ARDUINO

ABSTRACT

Use potentiometer to:

- Control LED ring on Renger (Auriga Shield).
- Regulate the speed of a fan, in addition to moving it towards the sides.

DIDACTIC OBJECTIVES

- Use Arduino language to program Ranger.
- For loop.
- Function map.

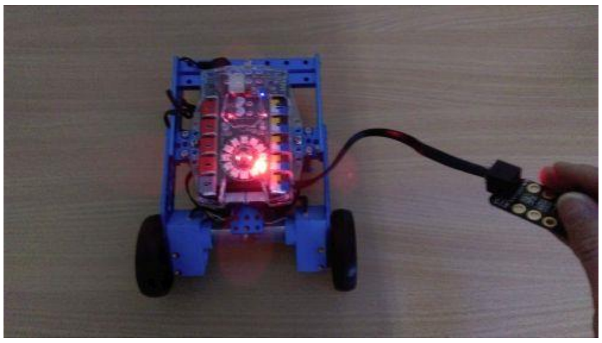
STEM Subject: Science☐ Technology ☐ Engineering☒ Mathematics☒

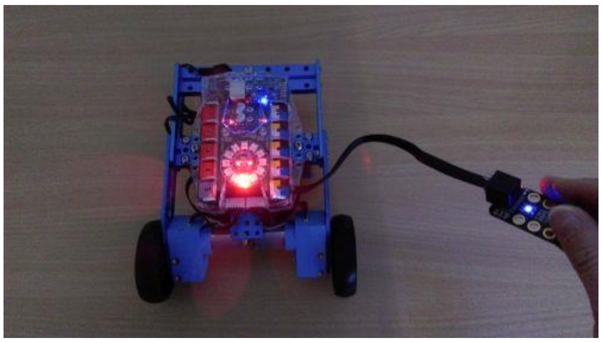
Education Level: 12-14 years☐ 14-16 years☒

PROBLEM STATEMENT

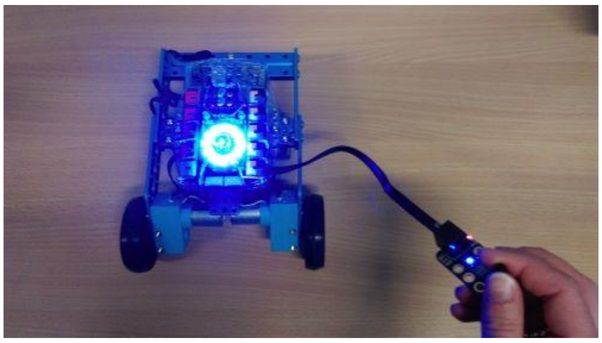
Some students do not know what is a potentiometer, so through this activity they will know more about it and will be able to apply it in different daily uses. We use Arduino language to write the program which control led ring on Renger.

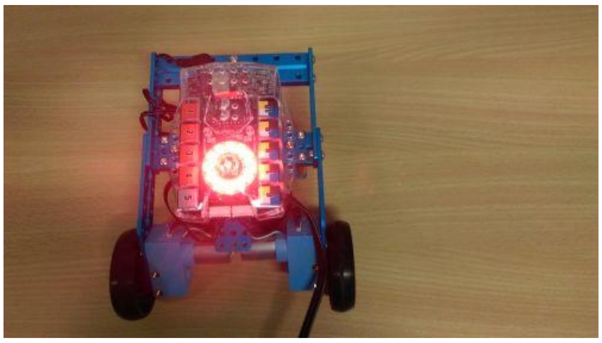
Proposition 1: At the same time only one led shines, but when user spins the potentiometer the next led shines





Proposition 2: All leds light in the same color, but the color is changing when the user spins the potentiometer

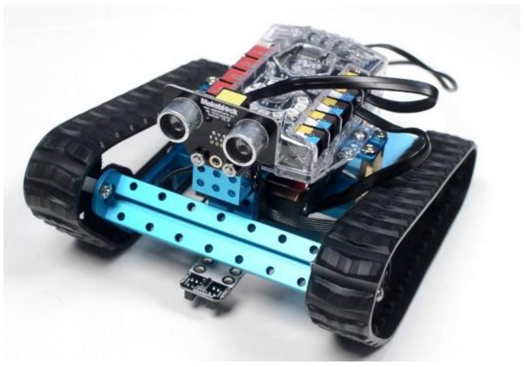




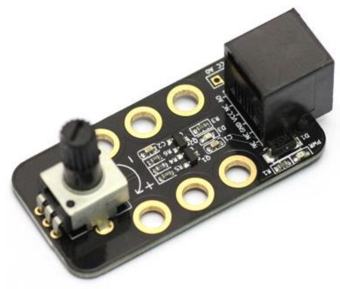
In the second part, we use potentiometer to control the fan. We can switch it on and off, as well as regulate the speed.

BOM (Bill Of Materials needed)

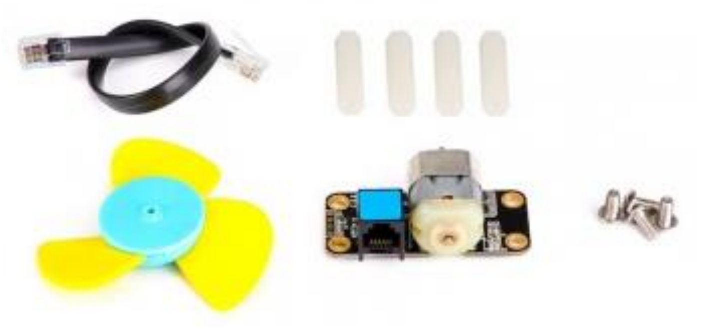
- The robot Ranger with led ring:



- Potentiometer sensor:



❖ Fan Pack:



❖ Ultrasonic Sensor:

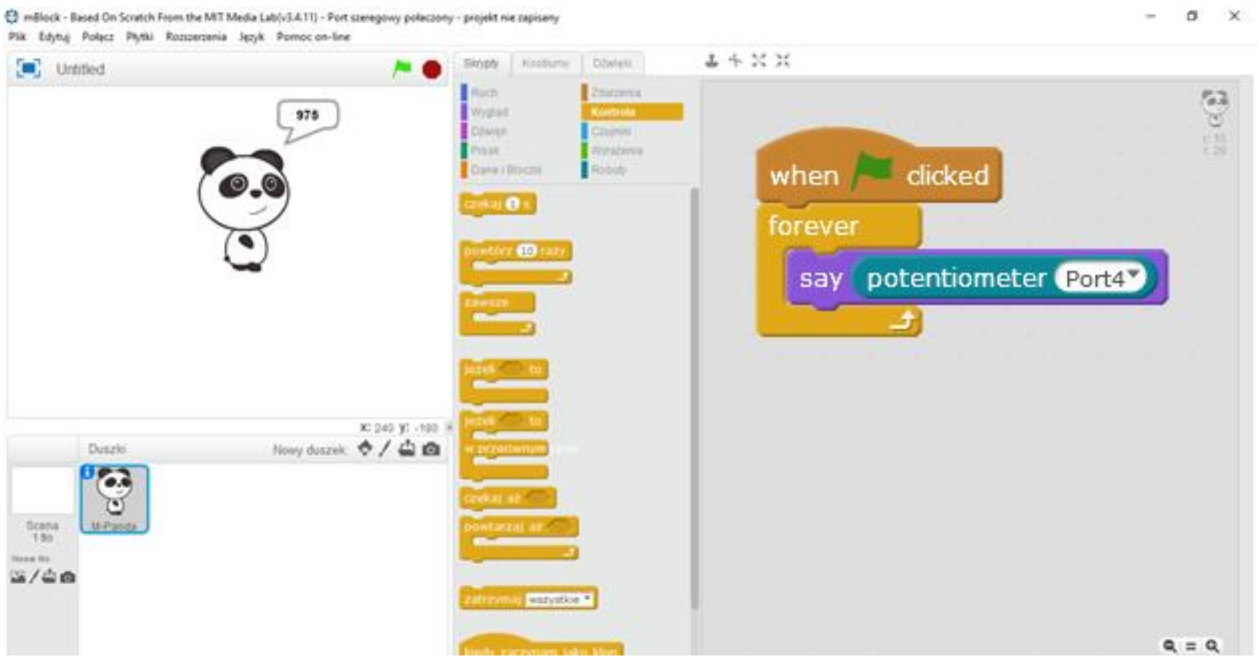
ELEMENT	ID	CABLE	AMOUNT	PORT 6				PORT 7				PORT 8				PORT 9				PORT 10				P.MOT1	P.MOT2
				Y	B	W	Bl	Y	B	W	Bl	Y	B	W	Bl	Y	B	W	Bl	Y	B	W	Bl	W*	W*
mBot Ranger			1																						
Motor 1	W*																							W*	
Motor 2	W*																								W*
Me RJ 25 adapter	Y																								
	B																								
	Bl																								
Mini Pan-Tilt kit																									
It has 2 servos.																									
We have to connect the servo to a RJ25 adapter																									
Mini Gripper																									
We have to connect the servo to a RJ25 adapter																									
Me 7-Segment serial display	B																								
Me Led Matrix 8x16	B																								
Me Ultrasonic sensor	Y	(1)	1	Y																					
Me Temperature Sensor - Waterproof	Y																								
Me Line Follower	B																								
Me Potentiometer sensor	Bl	(1)	1												Bl										
Me TFT LCD Screen	W																								
Me Sound sensor	Bl																								
Me Touch sensor	B																								
Mini Fan Pack	B	(1)	1																		B				
Me Temperature and Humidity sensor	Y																								
Me 130 Motor Fan Pack	B																								
RJ25 cables			3																						
Structures and beams																									
Laptops			1																						
Attrezzo (not essential)																									

ACTIVITY DESCRIPTION

First version

This is a 50k dial type potentiometer with know which can be rotated up to 270 degrees. It can convert rotary motion to an analog input which can be used to control the speed of a mobile robot, the brightness of RGB LEDs, or others.

Signal type is analog. Range from 0 to 970. It is easy to check it in mBlock program with mBot.

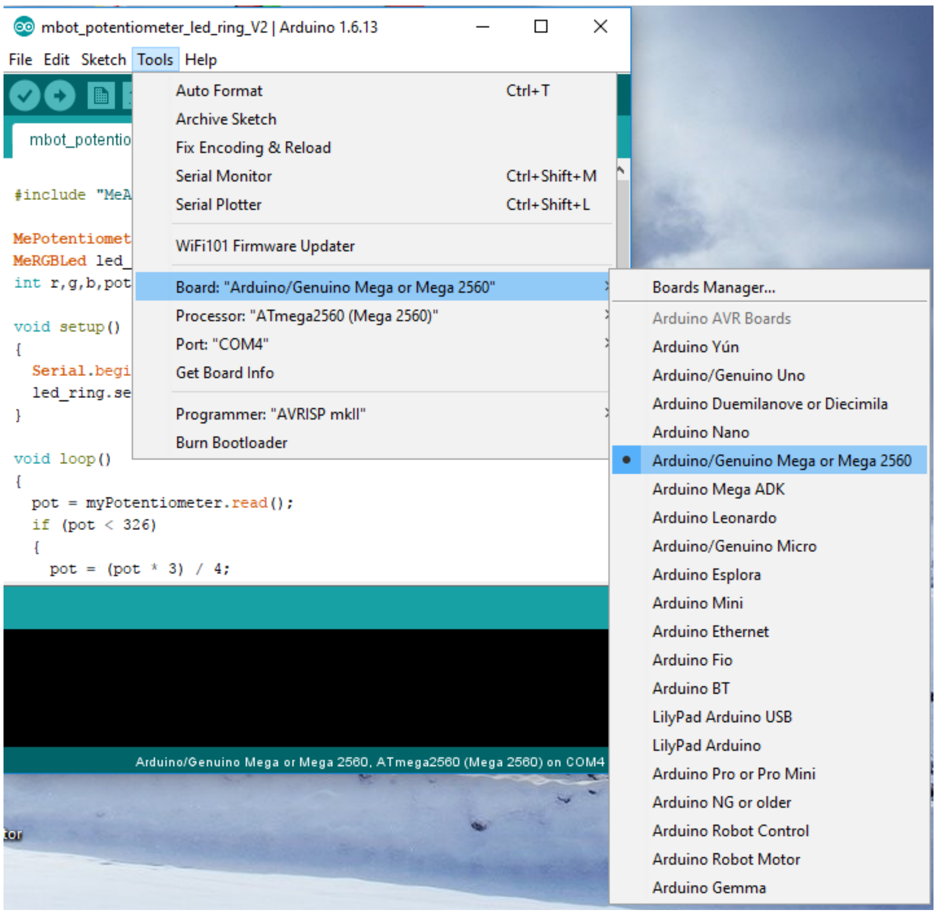

mBlock - Based On Scratch From the MIT Media Lab(v3.4.11) - Port szeregowy połączony - projekt nie zapisany
975

```
when [flag] clicked
forever
    say potentiometer Port4
```

Arduino

The aim of activity is to work with ranger in Arduino language.

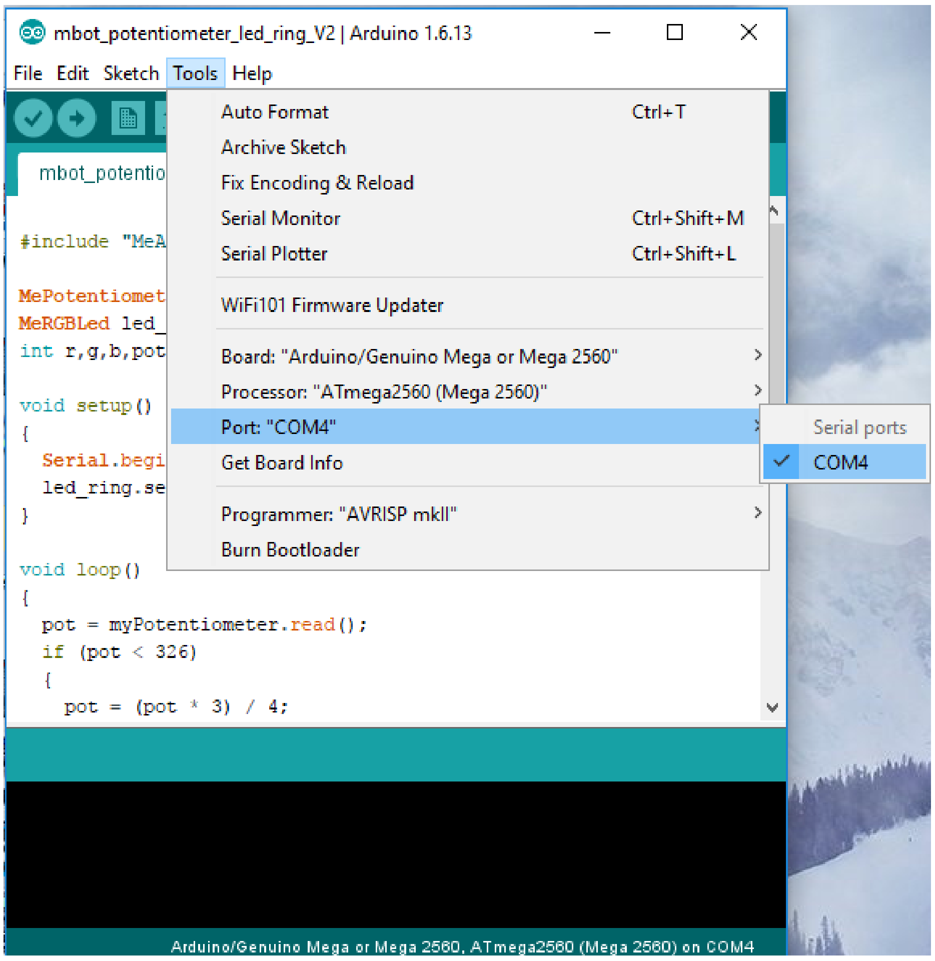
Working with ranger you should change the board to Mega 2560.



```
#include "MeA

MePotentiomet
MeRGBLed led_
int r,g,b,pot

void setup()
{
  Serial.begi
  led_ring.se
}

void loop()
{
  pot = myPotentiometer.read();
  if (pot < 326)
  {
    pot = (pot * 3) / 4;
```

```
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM4
```

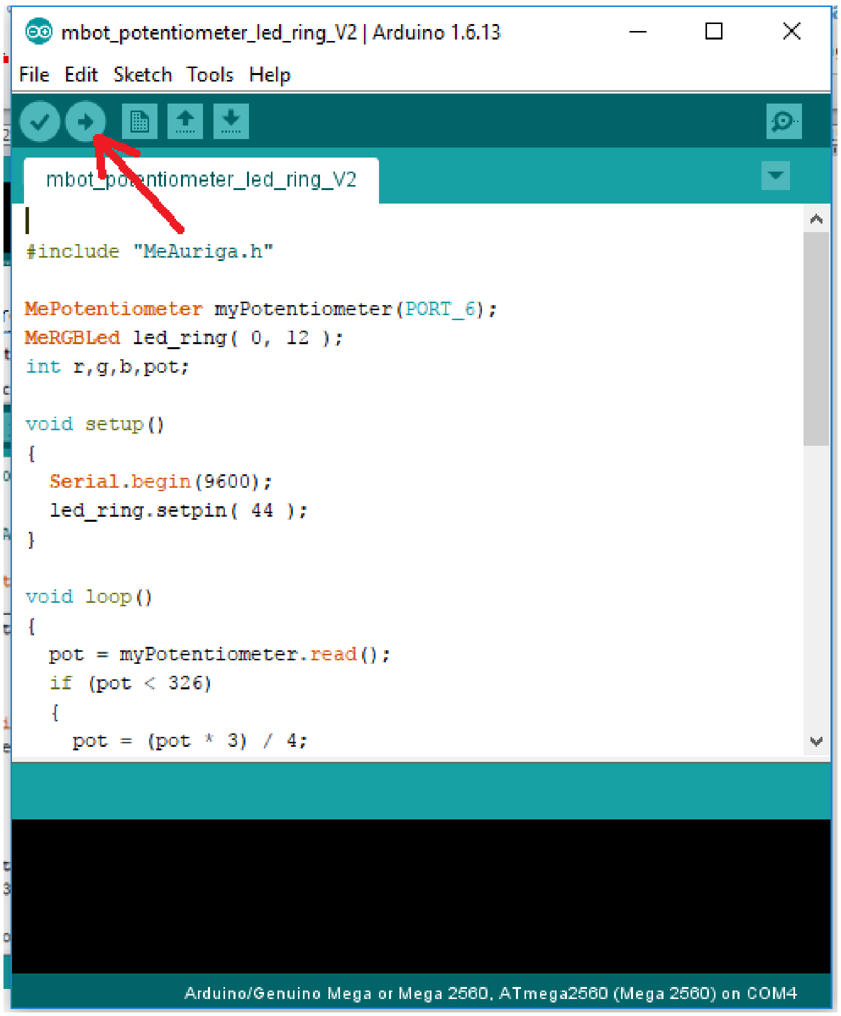
When you connect ranger to PC in the Tools menu you will find the COM port.


mbot_potentiometer_led_ring_V2 | Arduino 1.6.13
File
Edit
Sketch
Tools
Help
mbot_potentio
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Mega or Mega 2560"
Processor: "ATmega2560 (Mega 2560)"
Port: "COM4"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
✓ COM4

```
#include "MeA

MePotentiomet
MeRGBLed led_
int r,g,b,pot

void setup()
{
  Serial.begi
  led_ring.se
}

void loop()
{
  pot = myPotentiometer.read();
  if (pot < 326)
  {
    pot = (pot * 3) / 4;
```

```
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM4
```

When your program is ready click the button *Upload:*


mbot_potentiometer_led_ring_V2 | Arduino 1.6.13
File Edit Sketch Tools Help
mbot_potentiometer_led_ring_V2

```
#include "MeAuriga.h"

MePotentiometer myPotentiometer(PORT_6);
MeRGBLed led_ring( 0, 12 );
int r,g,b,pot;

void setup()
{
  Serial.begin(9600);
  led_ring.setpin( 44 );
}

void loop()
{
  pot = myPotentiometer.read();
  if (pot < 326)
  {
    pot = (pot * 3) / 4;
```

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM4

Writing the program to control led ring

First learn about method to control led ring.

There is fragment of documentation:

bool	setColorAt (uint8_t index, uint8_t red, uint8_t green, uint8_t blue)
	set the rgb value of the led with the index.

void	show ()
	become effective of all led's change.

The ring has 12 leds. They have numbers from 1 to 12.

To set colour of all leds we use instruction: `setColor(0, r, g, b)` – 0 means all leds. Then put three numbers from 0 to 255.

After each setColor instruction use show() and delay() to see the action.

For example:

```
#include "MeAuriga.h"

MeRGBLed led_ring( 0, 12 );

void setup()
{
  Serial.begin(9600);
  led_ring.setpin( 44 );
}

void loop()
{
  led_ring.setColor(0, 0, 0, 0);
  led_ring.show();
  delay (500);
  led_ring.setColor(2, 100, 0, 0);
  led_ring.show();
  delay(200);
  led_ring.setColor(5, 0, 0, 100);
  led_ring.show();
  delay(200);
  led_ring.setColor(12, 0, 100, 0);
  led_ring.show();
  delay(200);
  }
```

This program works as follow:

1. Turn off all leds.
2. Wait 500ms.
3. Light led number 2 with red colour.
4. Wait 200ms.
5. Light led number 5 with blue colour.

6. Wait 200ms.
7. Light led number 12 with green colour.
8. Wait 200ms.
9. Come back to step 1.

At the end of loop three leds light.

Use *for loop* to light all leds one by one:

```
#include "MeAuriga.h"

MeRGBLed led_ring( 1, 12 );

void setup()
{
  Serial.begin(9600);
  led_ring.setpin( 44 );
}

void loop()
{
 for (int i=1;i<=12;i++)
    {
      led_ring.setColor(0, 0, 0, 0);
      led_ring.show();
      delay (100);
      led_ring.setColor(i, 100, 0, 0);
      led_ring.show();
      delay(200);
    }
}
```

At one moment only one led is lighting because each iteration starts with instruction `setColor(0, 0, 0, 0)`.

When you miss it, for i=12 all leds will be lighting:

```
void loop()
{
  led_ring.setColor(0, 0, 0, 0);
  led_ring.show();
  delay (100);
  for (int i=1;i<=12;i++)
    {
      led_ring.setColor(i, 100, 0, 0);
      led_ring.show();
      delay(200);
    }
}
```

And the last version with for-loop.

```
void loop()
{
   for (int i=1;i<=12;i++)
    {
      led_ring.setColor(i, 100, 0, 0);
      led_ring.show();
      delay(200);
    }
   for (int i=1;i<=12;i++)
    {
      led_ring.setColor(i, 0, 0, 0);
      led_ring.show();
      delay(200);
    }
}
```

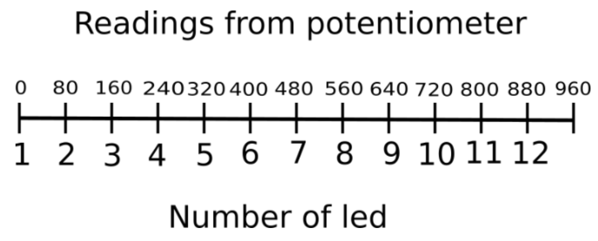
This time the second for-loop turns off the led one by one.

The map function

The potentiometer gives the number from 0 to 970, but we want to control 12 leds:

$$970 \div 12 \approx 80$$

Look at picture:



To transform readings from potentiometer which is analog to integer number we can use map function:

Syntax of function:

map(value, fromLow, fromHigh, toLow, toHigh)

Parameters

value: the number to map.

fromLow: the lower bound of the value's current range.

fromHigh: the upper bound of the value's current range.

toLow: the lower bound of the value's target range.

toHigh: the upper bound of the value's target range.

In program we just write:

```
mapped = map(myPotentiometer.read(), 0, 970, 1, 12);
```

The value mapped is the number of led.

The final program

```
#include "MeAuriga.h"

MePotentiometer myPotentiometer(PORT_6);

MeRGBLed led_ring( 0, 12 );

int mapped;


void setup()

{

  Serial.begin(9600);

  led_ring.setpin( 44 );

}


void loop()

{

  led_ring.setColor(0, 0, 0, 0);

  led_ring.show();

  mapped = map(myPotentiometer.read(), 0, 970, 1, 12);

  led_ring.setColor(mapped, 100, 0, 0);

  led_ring.show();

  delay(100);

}
```

Program to control the colour

The second version is that we want to control the colour of leds.

From range 0 to 970 we have to combine three values which give us the r,g,b values in instruction:

```
     setColor(0, r, g, b);
```

We will devide the readings from potentiometer into four sections:

Value `pot` is the reading from potentiometer, but we need to reduce it to values from 0 to 255

The readings	$pot \in \langle 0, 255)$	$pot \in \langle 255, 510)$	$pot \in \langle 510, 765)$	$pot \in \langle 765, 970)$
Reduction		$pot = pot - 255$	$pot = pot - 510$	$pot = pot - 765$
Pot after reduction	$pot \in \langle 0, 255)$	$pot \in \langle 0, 255)$	$pot \in \langle 0, 255)$	$pot \in \langle 0, 205)$
The definition of red	$r = 255 - pot$	$r = 1$	$r = pot$	$r = 255 - pot$
The definition of green	$g = pot$	$g = 255 - pot$	$g = 1$	$g = pot$
The definition of blue	$b = 1$	$b = pot$	$b = 255 - pot$	$b = 1$
The color for pot=0	(255,0,1)	(1,255,0)	(0,1,255)	(255,0,1)
The color for pot=125	(130,125,1)	(1,130,125)	(125,1,130)	(130,125,1)
The color for pot=254	(1,254,1)	(1,1,254)	(254,1,1)	(50,205,1)

As you see when value `pot` changes the interval the colour doesn't change:

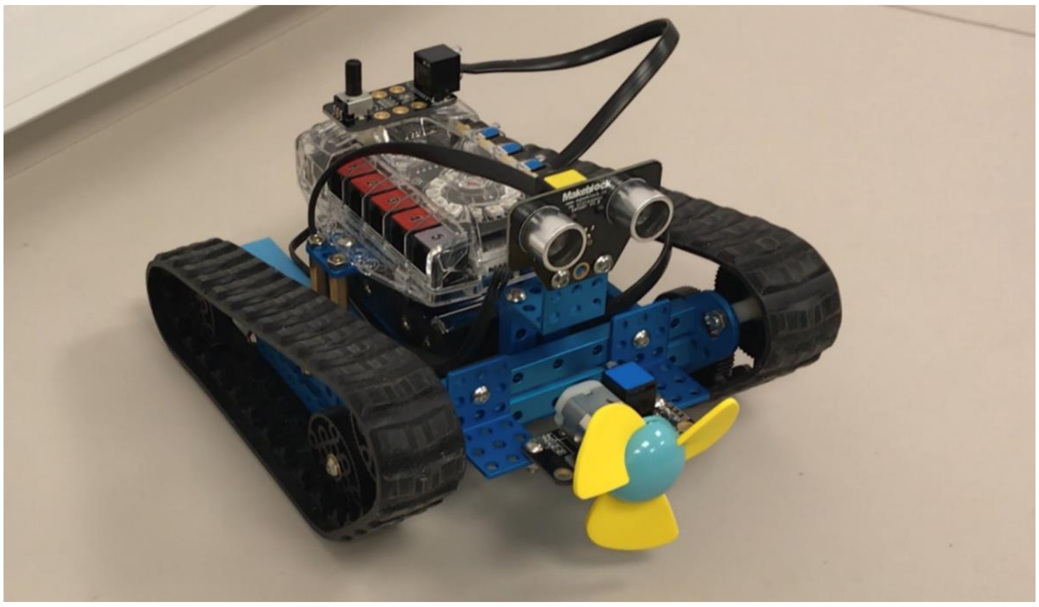
```
#include "MeAuriga.h"

MePotentiometer myPotentiometer(PORT_6);
MeRGBLed led_ring( 0, 12 );
int r,g,b,pot;

void setup()
{
  Serial.begin(9600);
  led_ring.setpin( 44 );
}

void loop()
{
  pot = myPotentiometer.read();
if (pot < 255)
  {
    r = 255 - pot;
    g = pot;
    b = 1;
  }
  else if (pot < 510)
  {
    pot = pot-255;
    r = 1;
    g = 255 - pot;
    b = pot;
  }
  else if (pot < 765)
  {
    pot = pot-510;

    r = 1;
    g = 255 - pot;
    b = pot;
  }
  else
  {
    pot = pot-765;

    r = pot;
    g = 1;
    b = 255 - pot;
  }
  led_ring.setColor(0, r, g, b);
  led_ring.show();
  delay(100);
}
```

Second version

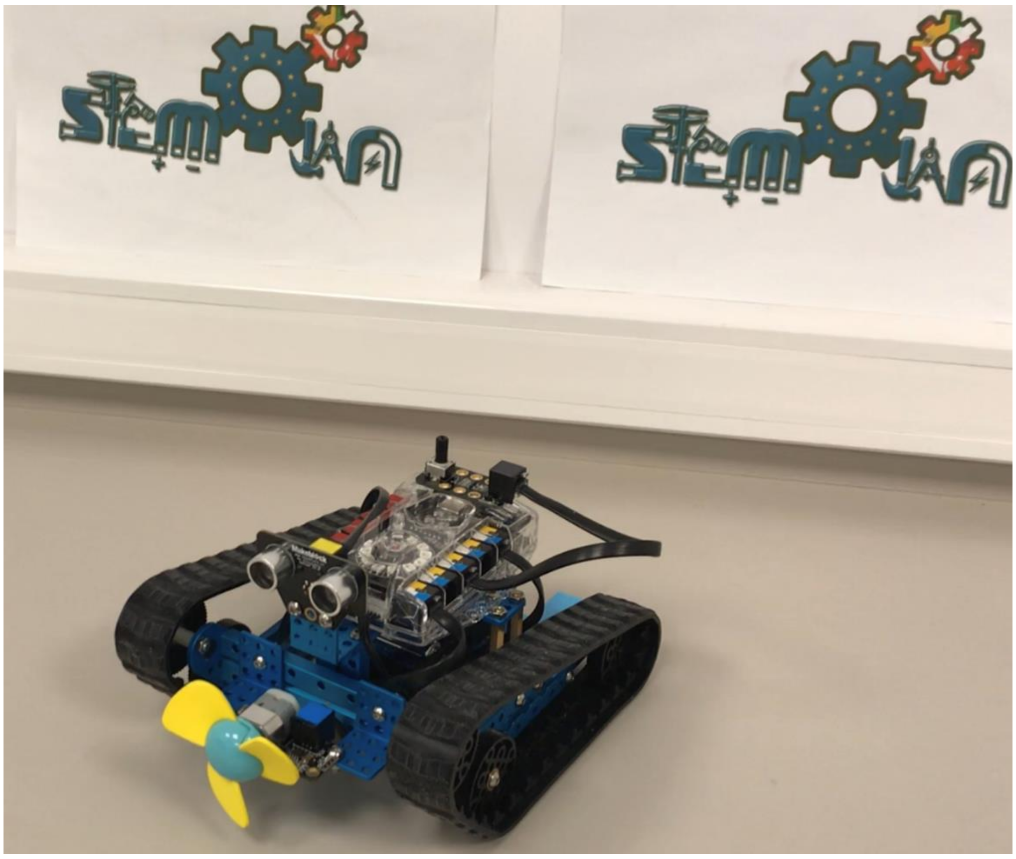
The Ranger, with the help of the Potentiometer, will act as a fan, and can switch it on and off, as well as regulate the speed.

The Makeblock fan, like most Arduino binary engines, only has two states, switch on and off, and it is not possible to regulate the speed. However, it can regulate at regular intervals of time, determined with the potentiometer, stop for a brief moment the engine, we will manage to regulate its speed accurately.

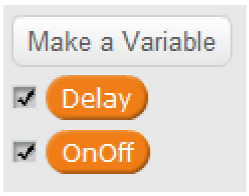
a. First, we connect the Potentiometer Sensor and Fan Pack to mBot Ranger.



b. And then, we programming the mBot Ranger:



We use these variables for:

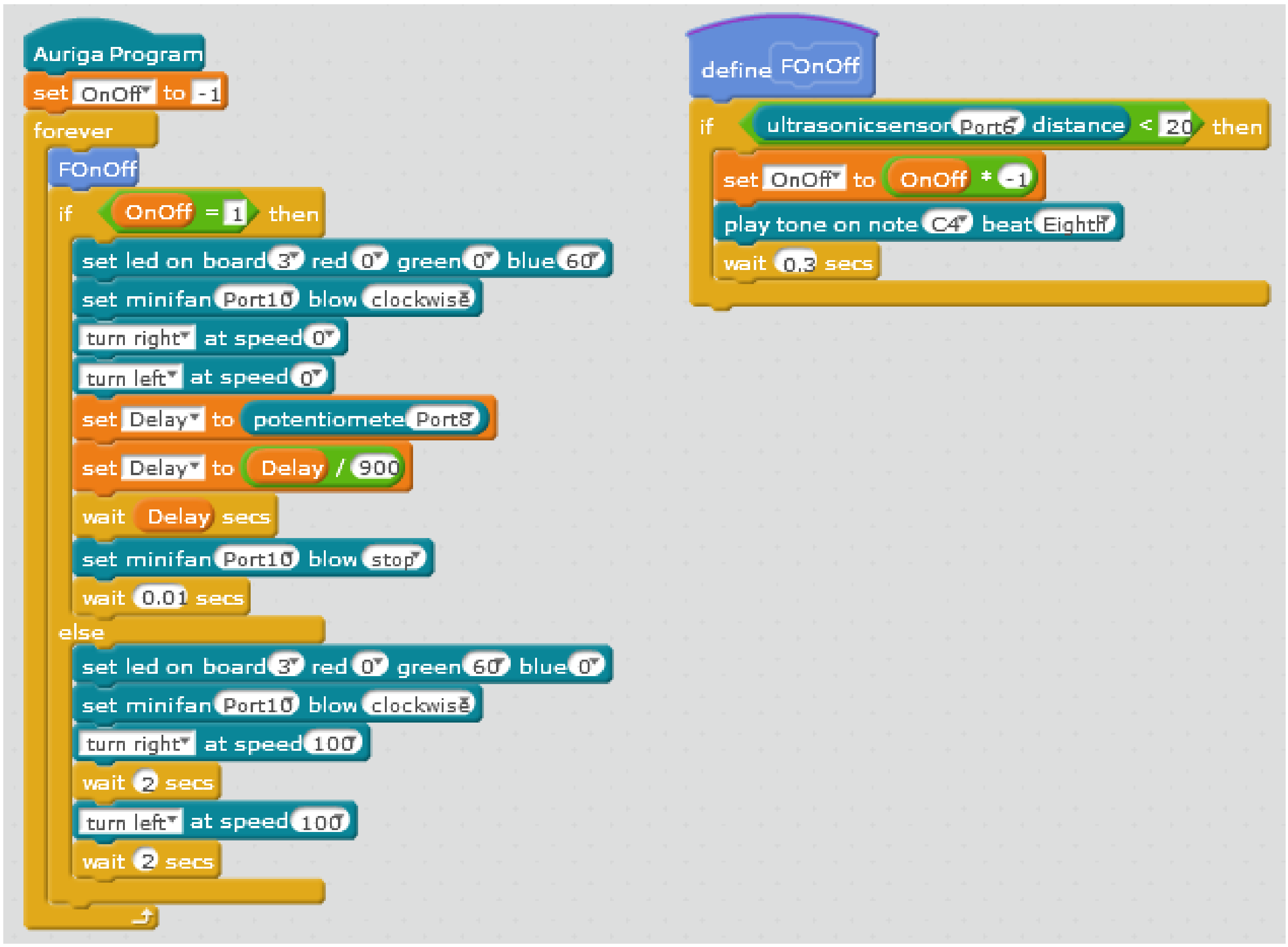


- Delay: Wait time.
- OnOff: If the fan is switch on or off.

We use these functions for:

- FOnOff: Function responsible for switching the fan off or on.

Once we know the variables that we need and the function, we will show the code of the complete program:

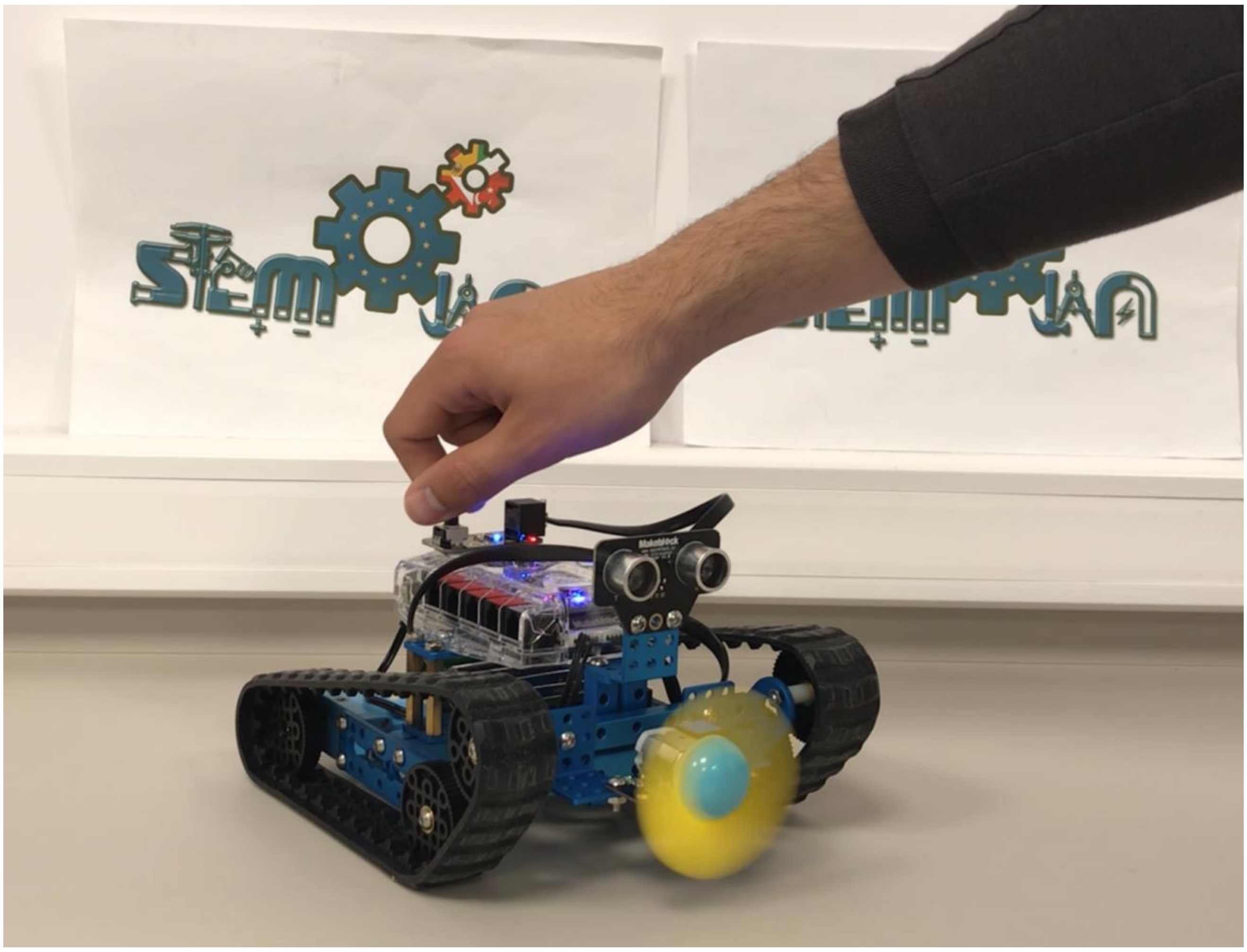


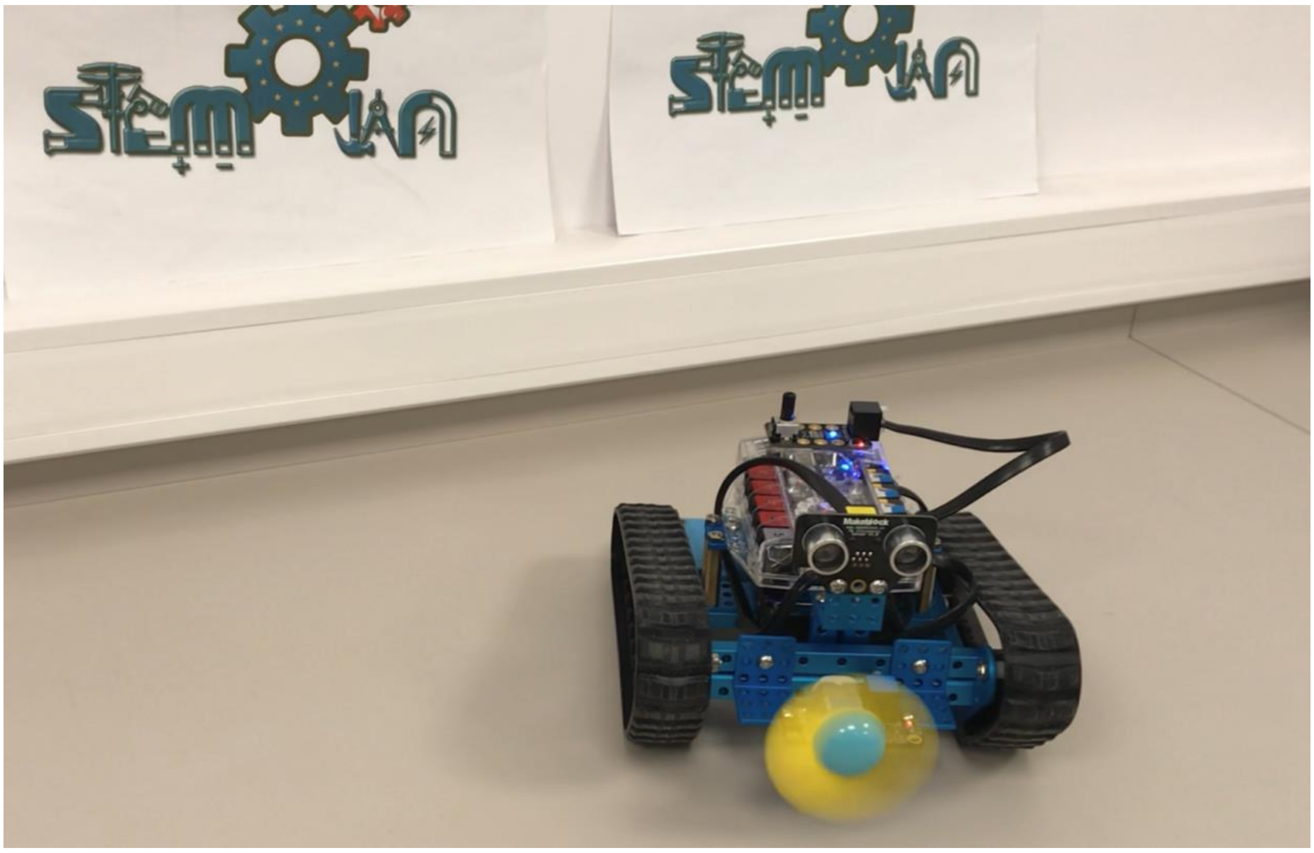
```
Auriga Program
set OnOff to -1
forever
    FOnOff
    if OnOff = 1 then
        set led on board 3 red 0 green 0 blue 60
        set minifan Port10 blow clockwise
        turn right at speed 0
        turn left at speed 0
        set Delay to potentiomete Port8
        set Delay to Delay / 900
        wait Delay secs
        set minifan Port10 blow stop
        wait 0.01 secs
    else
        set led on board 3 red 0 green 60 blue 0
        set minifan Port10 blow clockwise
        turn right at speed 100
        wait 2 secs
        turn left at speed 100
        wait 2 secs
```

```
define FOnOff
if ultrasonicsensor Port6 distance < 20 then
    set OnOff to OnOff * -1
    play tone on note C4 beat Eighth
    wait 0.3 secs
```

The loop of the program will consult in the "FOnOff" function the value provided by the potentiometer, as this value increases or decreases the fan speed will be altered.

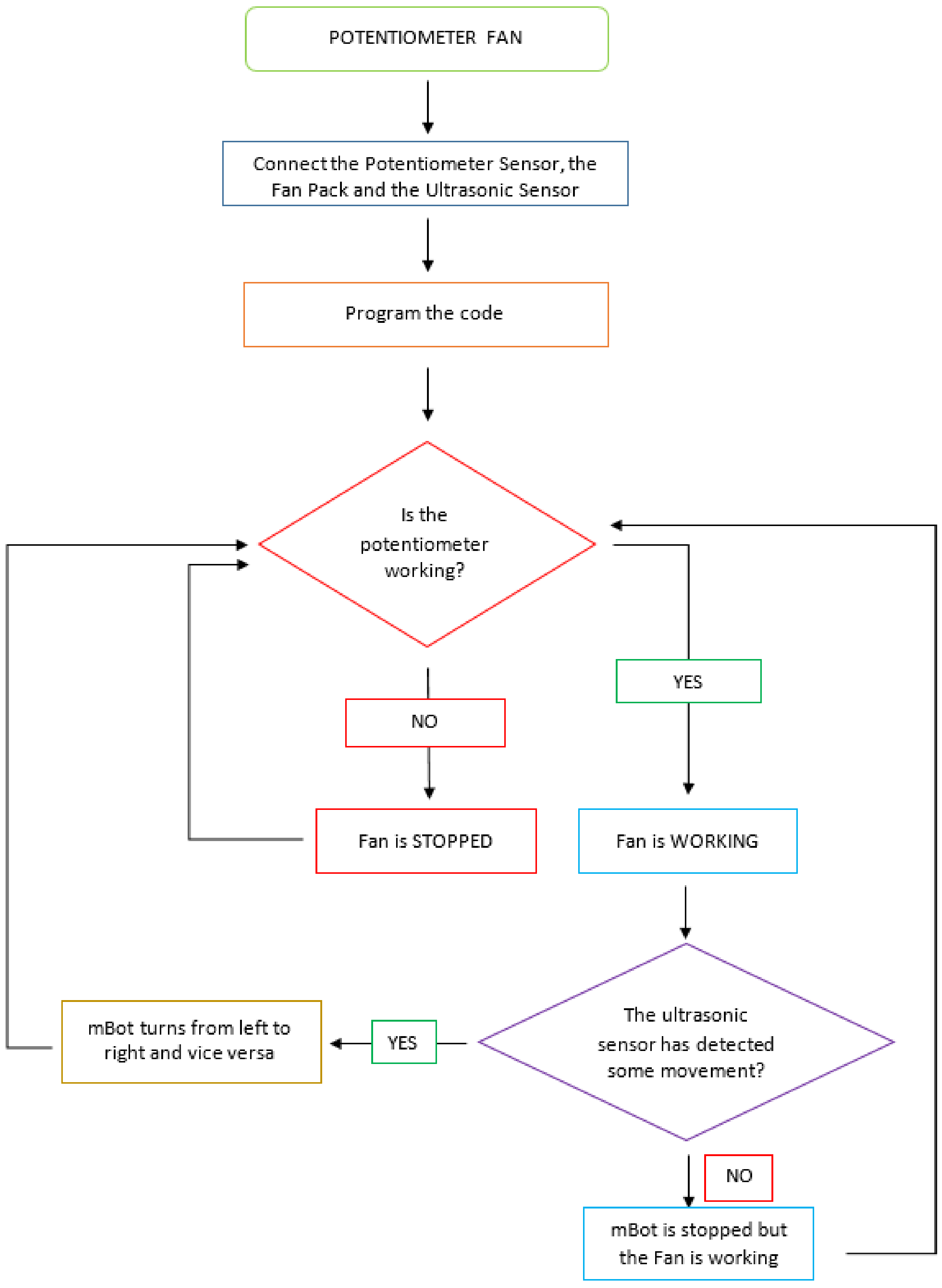
To make the activity more attractive, we have incorporated an ultrasonic sensor that when you put your hand or other object near it, the engines of the Ranger wheels will activate, which will start to rotate as if it were a standing fan.


STEM JAM
Makeblock


STEM JAM
Makeblock

FLOW CHART

Second version


POTENTIOMETER FAN
Connect the Potentiometer Sensor, the Fan Pack and the Ultrasonic Sensor
Program the code
Is the potentiometer working?
NO
Fan is STOPPED
YES
Fan is WORKING
The ultrasonic sensor has detected some movement?
YES
mBot turns from left to right and vice versa
NO
mBot is stopped but the Fan is working

STUDENT'S EVALUATION

After the activity student can modify the program. They can control the led ring in for loop.

In the last activity the colour of first and last interval are the same. Try to modify the program and devide it to 3 intervals.

BIBLIOGRAPHY

https://github.com/Makeblock-official/Makeblock-Libraries

http://wiki.makeblock.cc/library/docs/class_me_r_g_b_led.html

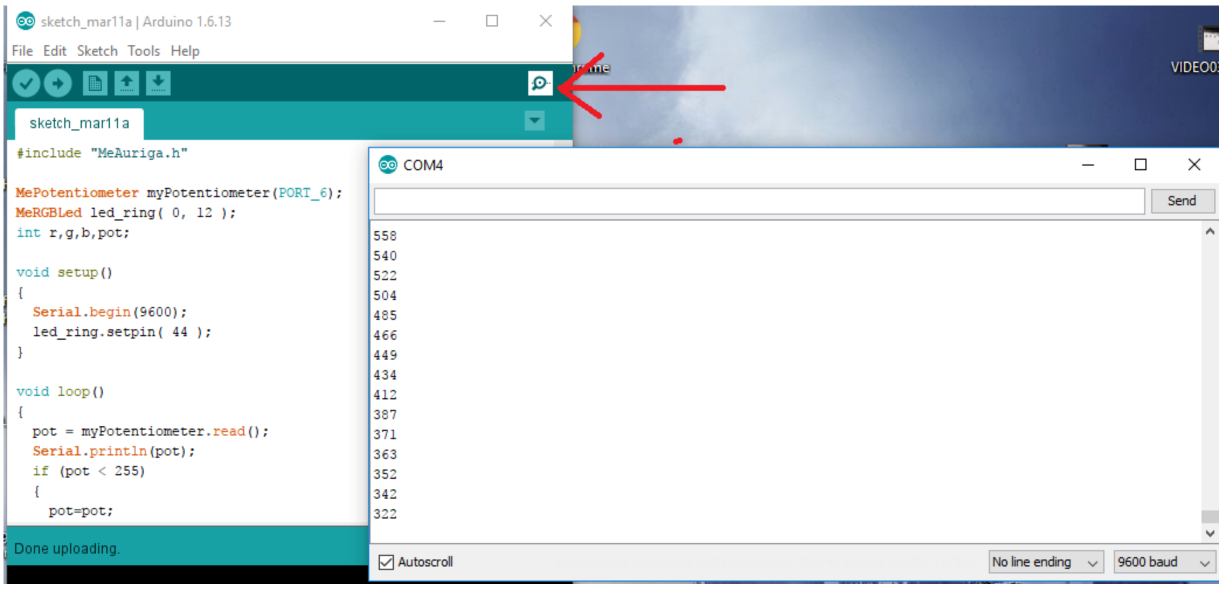
MORE INFORMATION

To show the readings from potentiometer or other sensor you can use the instruction:

Serial.println(k)

Where k is the reading form sensor.

Put this instruction in loop() section and click here:



```
#include "MeAuriga.h"

MePotentiometer myPotentiometer(PORT_6);
MeRGBLed led_ring( 0, 12 );
int r,g,b,pot;

void setup()
{
  Serial.begin(9600);
  led_ring.setpin( 44 );
}

void loop()
{
  pot = myPotentiometer.read();
  Serial.println(pot);
  if (pot < 255)
  {
    pot=pot;
```

```
558
540
522
504
485
466
449
434
412
387
371
363
352
342
322
```

☑ Autoscroll   No line ending   9600 baud