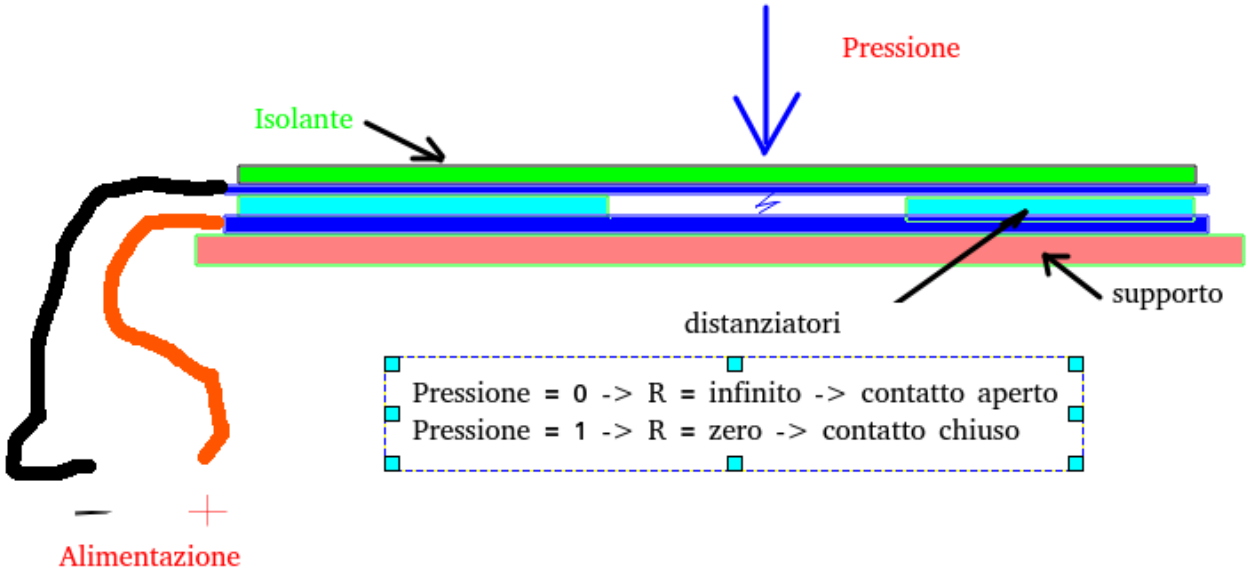
QBO.one Touch sensors

The touch sensors can operate in accordance with an electrically detectable physical change due to contact with a monitored surface.

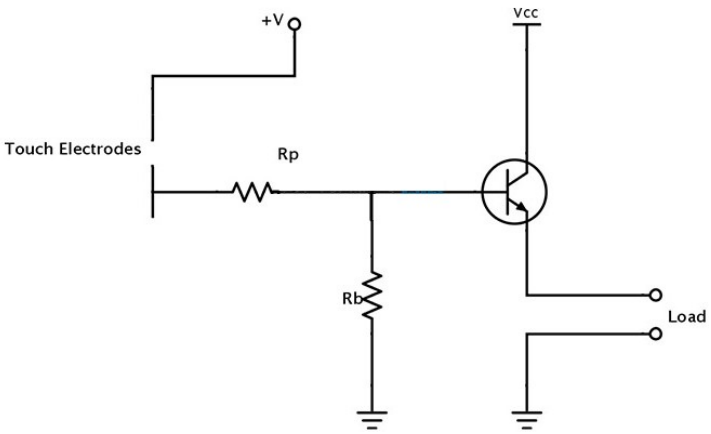
Basically the touch or tactile sensors can be due to a resistive variation or a capacitive variation of the detection system.

The detection system for resistive tactile sensors is usually obtained by means of spaced surfaces which, upon coming into contact with each other under the effect of a pressure, modify their resistive value, as if it were an ON / OFF switch.

The following figure shows the principle of operation schematically.

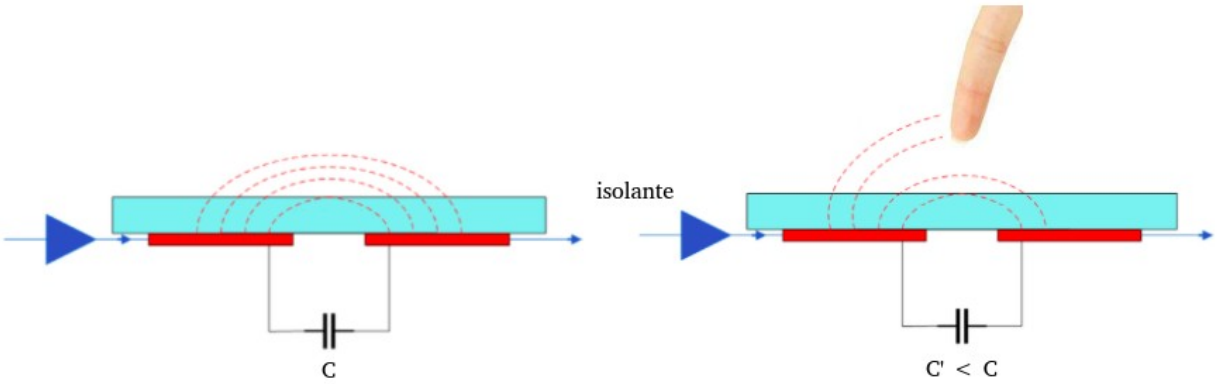


An electronic circuit that can detect this type of signal can be exemplified as in the figure below:



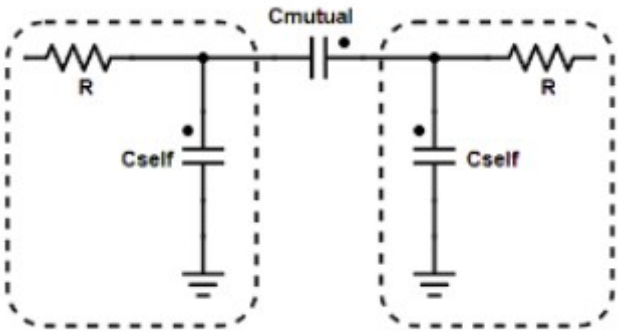
Another system for touch detection can be obtained through a capacitive sensor, such as those installed in the side and front blue plastics of QBO.One.

The operating principle in this case occurs as shown in the following figure:



The operating principle is that of reciprocal capacitive sensors, where the variation of the electric field produced by the parallel plates is obtained in relation to the approach of a foreign body, such as a finger, which by capturing lines of force, reduces the value ti this field which corresponds to a reduction of the "mutual capacity" between the two plates.

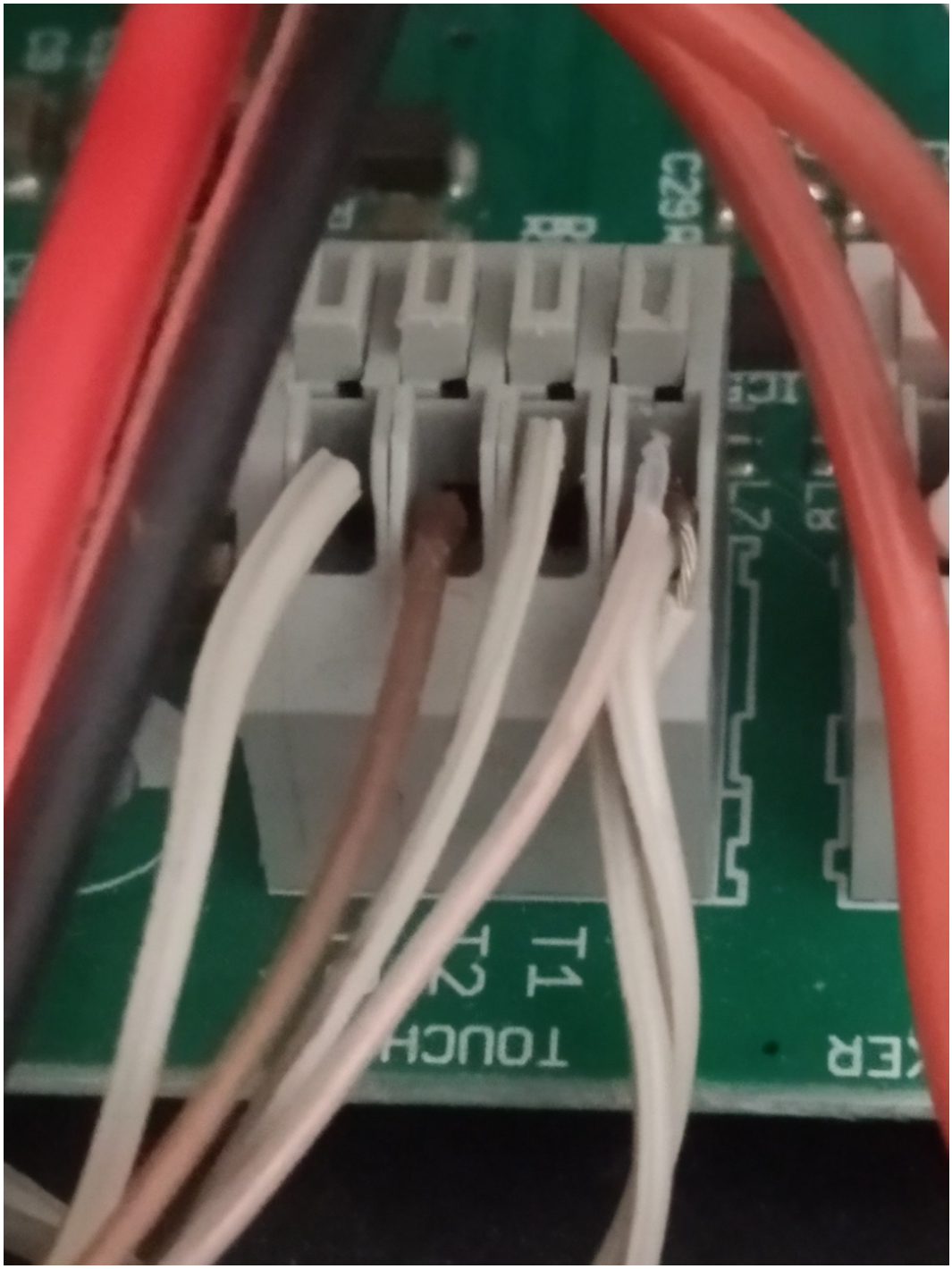
Schematically, an equivalent circuit could be represented as follows:



For mutual capacitance sensors, the sensing function is performed by an IC that drives a Tx electrode and detects the charge on an Rx electrode. The amount of charge measured on the Rx electrode is proportional to the mutual capacity between the two electrodes.

Electrical connections

The electrical connections in the QBO board. board are those shown in the figure:

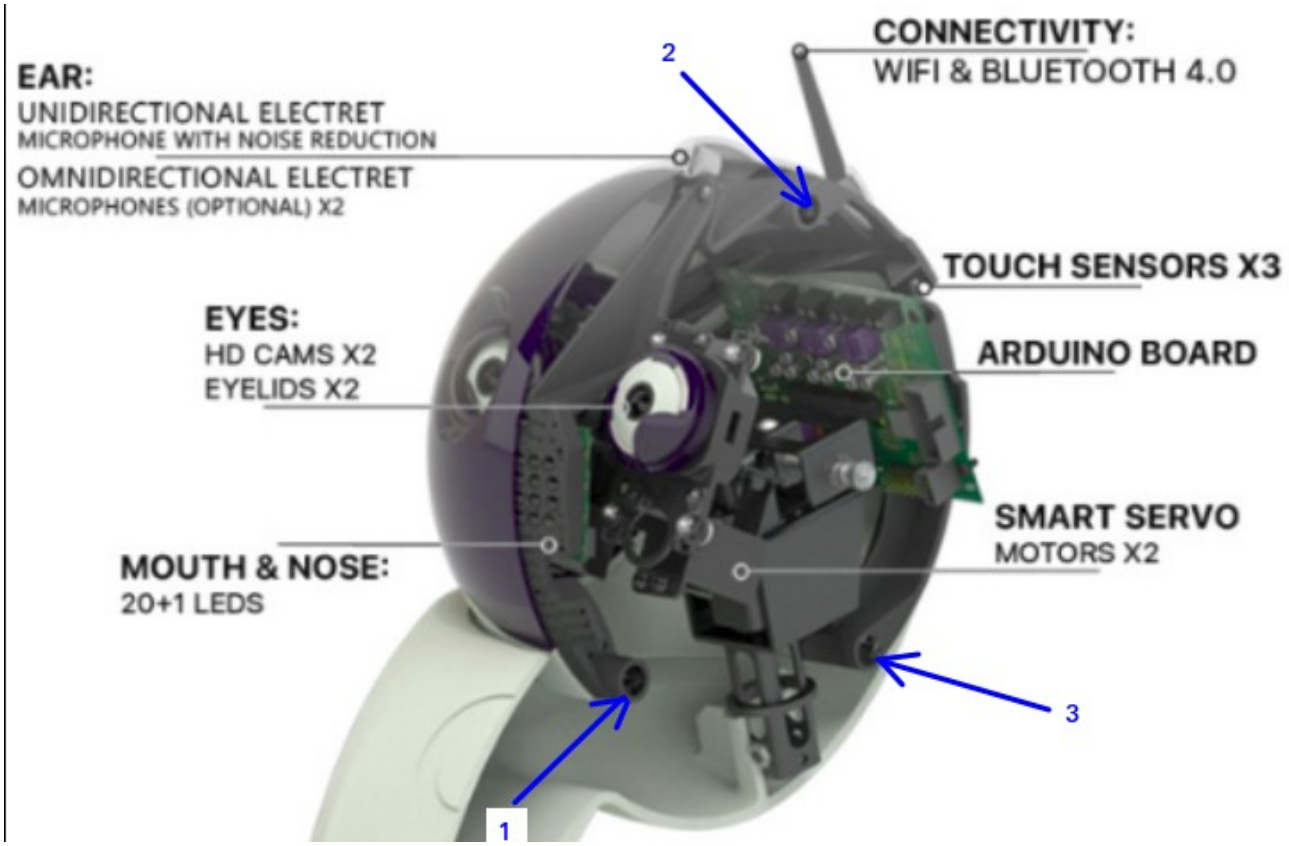


Where it can be seen that the common of the three signals are inserted in slot T0,

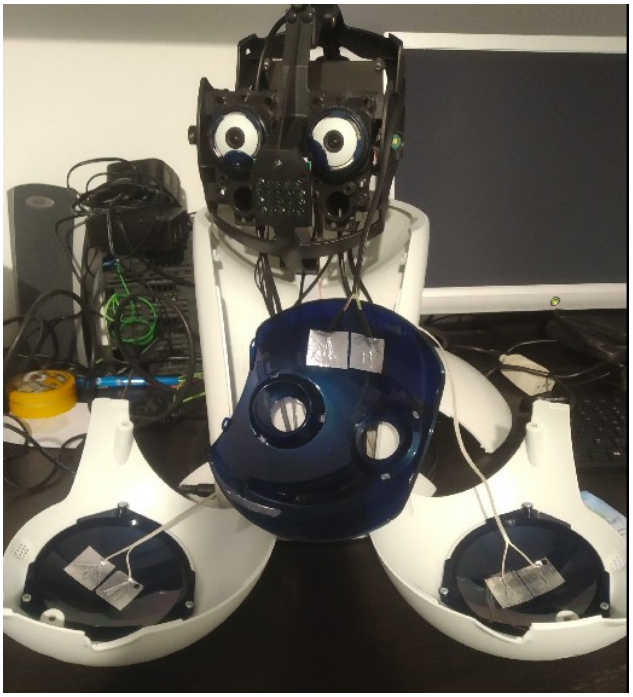
in T1 the strand of one of the plates of the left sensor is inserted, in T2, the strand of the signal of the central sensor is inserted and in T3, the signal of the right sensor is inserted.

To access the aluminum plates installed in the plastics, it is necessary to remove the screws that join the right and left parts of the QBO.One head, using the appropriate Allen key.

The screws are inserted as shown in the figure:



The connections to the plates are made as shown in the following figure:



The plates consist of an adhesive aluminum film having approximately the size of a 25 x 25 mm square.

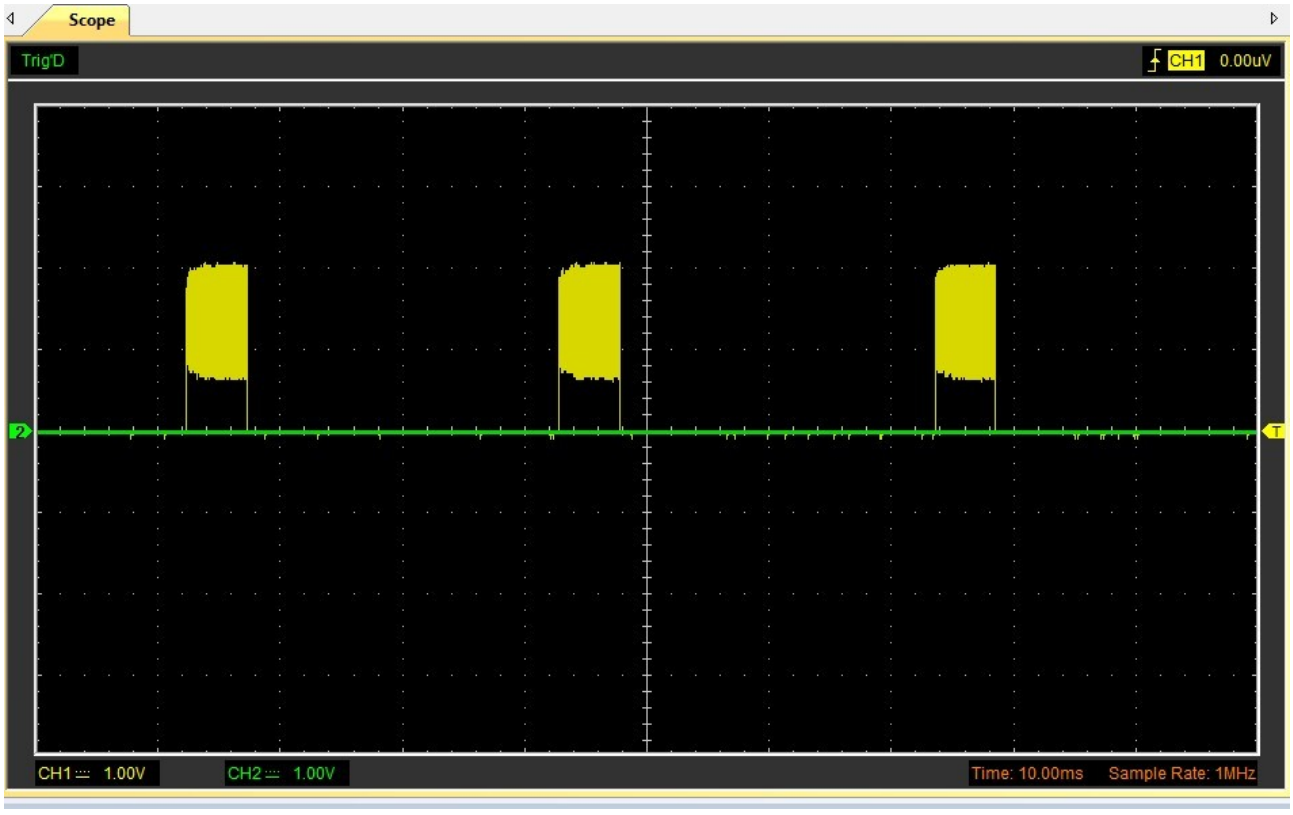
If you want to measure the electrical resistance and the voltage between the plates "A" and "B" making up the sensors, you should detect values similar to those shown in the table:

A	B	C	D
RIGHT	CENTRAL	LEFT	tester ASITA
R_A_B = 3.66 K	R_A_B = 2.85 K	R_A_B = 3.70 K	rosso = com
R_B_A = 10.05K	R_B_A = 11.21 K	R_B_A = 9.40K	A- nero = com
Vdc(B-A)= 0.615 V	Vdc(B-A)=0.265V	Vdc(B-A) = 0.225 V	Nero = com

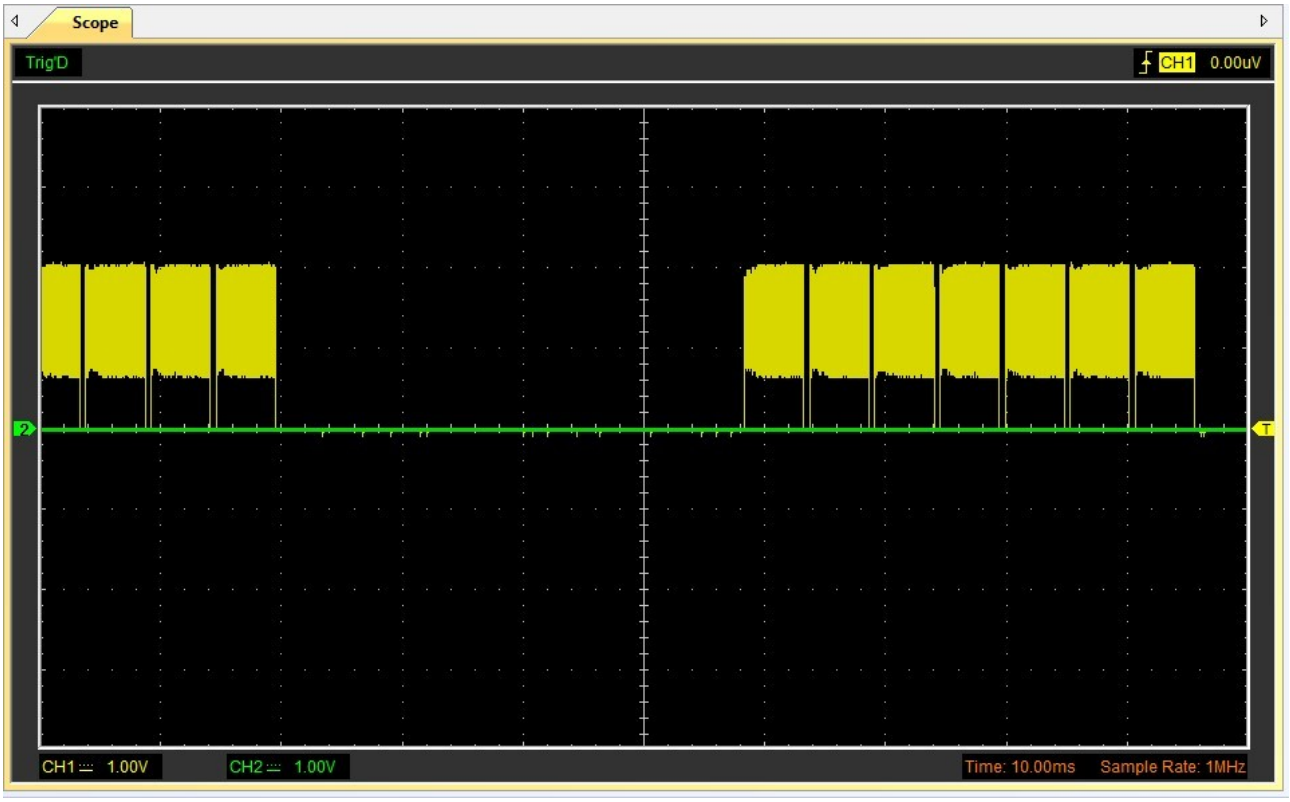
Note that the resistance measurement varies according to the polarization.

Further measurements could be conducted using an oscilloscope, obtaining results similar to those shown in the following illustrations:

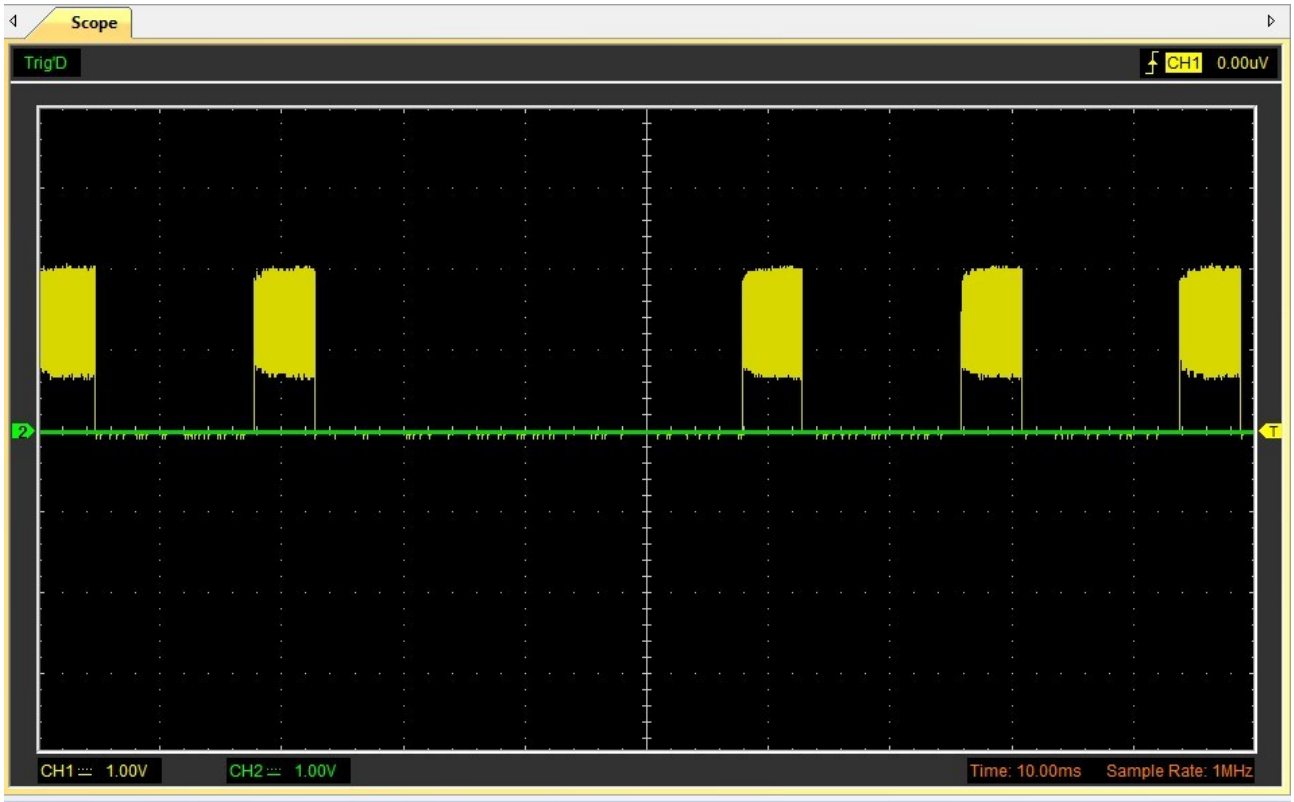
This illustration shows the wave function for the left sensor:



This, for the right sensor:



and finally, this for the central sensor:



Below is an example for the test program built in python:

```
#!/usr/bin/env python2
# -*- coding: latin-1 -*-

#import os
#import errno
import serial
import sys
import time
import QboCmd

port = '/dev/serial0'
move_to = [0]
touch_str = ''

try:

	print(port)

	# Open serial port
	ser = serial.Serial(port, baudrate=115200, bytesize=serial.EIGHTBITS, stopbits=serial.STOPBITS_ONE, parity=serial.PARITY_NONE, rtscts=False, dsrdtr=False, timeout=0)

	print "Open serial port sucessfully."
	print(ser.name)

	QBO = QboCmd.Controller(ser)

except:
	print "Error opening serial port."
	sys.exit()


def WaitForTouch():


	touch_str = ""
	touch = QBO.GetHeadCmd("GET_TOUCH", 0)

	if touch:
		global touch_str

		if touch == [1]:
			move_to = [1]
			touch_str = "Touch: right"
			print("Right Positon")
			print("Blue Nose")
			QBO.SetNoseColor(1)
			time.sleep(1)
			QBO.SetNoseColor(0)
```

```
                elif touch == [2]:
                        touch_str = "Touch: up"
                        print("Center head")
                        print("Cyan Nose")
                        QBO.SetNoseColor(5)
                        print("Smile")
                        QBO.SetMouth(0x110E00) #happy
                        QBO.SetServo(1,500, 100)#Axis,Angle,Speed
                        #Pause
                        time.sleep(1)
                        print("Switch off")
                        QBO.SetNoseColor(0)

                elif touch == [3]:
                        touch_str = "Touch: left"
                        print("Left position")
                        print("Green Nose")
                        QBO.SetNoseColor(6)
                        time.sleep(1)
                        QBO.SetNoseColor(0)

                if touch == [1] or touch == [2] or touch == [3]:
                        print("touch ",touch)
                        #fifo = os.open(FIFO_feel, os.O_WRONLY)
                        #os.write(fifo, touch_str)

        time.sleep(.250)
        return touch_str

def Move_Head():
        global touch_str

        #print("Head move_to..... ", touch_str)
        if touch_str == "Touch: right":
                #Move the head to the right if touch [1]
                print("Move right")
                QBO.SetServo(1,210, 100)#Axis,Angle,Speed
                time.sleep(1)
                print("Smile")
                QBO.SetMouth(0x110E00) #happy

                touch_str = 'ok'

        if touch_str == "Touch: left":
                #Move the head to the left if touch [3]
                print("Move left")
                QBO.SetServo(1,725, 100)#Axis,Angle,Speed
                #Pause
                time.sleep(1)
                print("Smile")
                QBO.SetMouth(0x110E00) #happy
```

```
                touch_str = 'ok'

def Move_ok():
        time.sleep(3)
        global touch_str
        if touch_str == "ok":
                print("Move center")
                QBO.SetServo(1,500, 100)#Axis,Angle,Speed
                time.sleep(3)
                print("by / by")
                QBO.SetMouth(0) #none
                time.sleep(1)
                QBO.SetNoseColor(0) # none
                touch_str = ''
                time.sleep(1)

while True:
        time.sleep(1)
        WaitForTouch()
        Move_Head()
        Move_ok()
```

Warning: this software has been created only for testing, on a special directory where QboCmd.py is also present. (the management software of the Qbo. Board interface, where, among other things, the monitoring data of the sensors are set as reported by the following program lines:

…
…

```
class Controller:
    def __init__(self, serial):
        self.port = serial
        self.port.timeout = 1.0

    INPUT_FLAG =  0xFF
    OUTPUT_FLAG = 0xFE
    INPUT_ESCAPE = 0xFD
    GET_VERSION = 0x40
    SET_MOUTH_VALUE = 0x44
    SET_STATE = 0x45
    SET_SERVO = 0x53
    SET_SERVO_PID = 0x50
    SET_SERVO_ANGLE = 0x71
    SET_SERVO_SPEED = 0x72
    SET_SERVO_ANGLE_REL = 0x73


# Commands definions :('NAME',value, number of set params, number of get params)

    cmd_params = {
        "GET_VERSION": (0x40, 0, 1),
        "SET_MOUTH_VALUE": (0x44, 4, 0),
```

```
    "SET_STATE": (0x45, 1, 0),
    "GET_TOUCH": (0x46, 0, 1),
    "SET_TOUC_PARAMS": (0x47,3,0),
    "SET_MIC_INPUT": (0x4A, 1, 0),
    "GET_MIC_REPORT": (0x4B, 0, 6),
    "SET_SERVO_CW_LIM":         (0x4C, 3, 0),
    "SET_SERVO_CCW_LIM": (0x4D, 3, 0),
    "SET_SERVO_ID": (0x4E, 2, 0),
    "SET_SERVO_BAUD": (0x4F, 2, 0),
    "SET_SERVO_PID": (0x50, 4 , 0),
    "SET_SERVO_LED": (0x51, 2, 0),
    "RESET_SERVO": (0x52, 1, 0),
    "SET_SERVO": (0x53, 5, 0),
    "SET_SERVO_ANGLE_REL": (0x73, 3, 0),
    "SET_SERVO_ENABLE": (0x54,2, 0),
    "GET_SERVO_SPEED": (0x56, 1, 2),
    "GET_SERVO_LOAD": (0x57, 1, 2),
    "GET_SERVO_TEMP": (0x58, 1, 1),
    "GET_SERVO_VOLTAGE": (0x59, 1, 1),
    "SERVO_ISMOVING": (0x5a, 1, 1),
    "GET_HARDWARE_ERROR": (0x5b, 1, 1),
    "GET_HEAD_SERVOS": (0x5C, 0, 4),
    "GET_SERVO_POSITION": (0x5D, 1, 2),
    "GET_SERVO_CW_LIM": (0x5E, 1, 2),
    "GET_SERVO_CCW_LIM": (0x5F, 1, 2),
    "GET_SERVO_BYTE_REG": (0x6F, 2, 1),
    "GET_SERVO_WORD_REG": (0x70, 2, 2),
    "SET_SERVO_ANGLE": (0x71,3,0),
    "SET_SERVO_SPEED": (0x72,3,0),
    "SET_ADC_REF": (0x81, 1, 0),
    "SET_DAC_REF": (0x82, 1, 0),
    "SET_ADC_OFFSET": (0x83, 6, 0),
    "SET_SERVO_OVERCURRENT": (0x84, 1, 0),
    "SET_SERVO_POWER": (0x85, 2, 0),
    "SET_ENABLE_SPEAKER": (0x86, 1, 0),
    "SET_MEAN_RMS": (0x87, 6, 0),
    "SET_USB2SERVO_FWD": (0x88, 1, 0),
    "RESET_SOUND": (0x8F, 5, 0)}

  touch_sampletime = {
    "320us": 0,
    "640us": 1 << 2,
    "1280us": 2 << 2,
    "2560us": 3 << 2}

  touch_average = {
    "SAMPLES 1": 0,
    "SAMPLES 2": 1 << 4,
    "SAMPLES 4": 2 << 4,
    "SAMPLES 8": 3 << 4,
    "SAMPLES 16": 4 << 4,
    "SAMPLES 32": 5 << 4,
    "SAMPLES 64": 6 << 4,
    "SAMPLES 128": 7 << 4}

  touch_cycletime = {
    "35 ms": 0,
    "70 ms": 1,
    "105 ms" : 2,
```

```
    "140 ms" : 3}

touch_channels = {
    "Channel none": 0,
    "Channel 1": 1,
    "Channel 2": 1 << 2,
    "Channel 3": 1 << 3,
    "Channel all": 7}
```

…
…